robotron

Anleitung für den
Programmierer

FORTRAN77

Benutzungshinweise

Arbeitsplatzcomputer A 7100 Betriebssystem SCP 1700

SYSTEMUNTERLAGEN- DOKUMENTATION 1/87	FOR77	MOS SCP 1700

MOS

Anleitung für den Programmierer

FOR77

Benutzungshinweise

SCP 1700

VEB Robotron-Projekt Dresden

C 1015-0200-2 M3030

Kurzreferat

Die vorliegende Anleitung für den Programmierer enthält die erforderlichen Informationen für die Nutzung des FORTRAN77-Compilers und der zugehörigen Laufzeitbibliotheken im SCP. Sie beschreibt die notwendigen Kommandos zum Übersetzen, Assemblieren, Verbinden und Ausführen von FORTRAN77-Programmen und spezifiziert alle implementationsabhängigen Elemente FOR77.
Sie liefert weiterhin Informationen zum Einbinden von externen Assemblerroutinen und zur Fehlerdiagnose. Als Kurzbezeichnung für die Gesamtheit des FORTRAN77-Compilers und der dazugehörigen Laufzeitbibliotheken im SCP wird FOR77 verwendet.
Die in dieser Dokumentation beschriebene Optimierung von FORTRAN77-Programmen ist vorbereitet, wird aber erst in einer späteren Version von FOR77 realisiert.
Aktuelle Informationen über Einschränkungen und Erweiterungen können der Datei F77INFO.TXT (Bibliotheksdiskette) entnommen werden.

Die vorliegende Systemunterlagen-Dokumentation,
FOR77, Benutzungshinweise,
entspricht dem Stand von 01/87 und ist ab Version 1.0 gültig.

Die Ausarbeitung der Unterlagen erfolgte durch
ein Kollektiv des VEB Robotron-Projekt Dresden.

Nachdruck, jegliche Vervielfältigung daraus sind unzulässig.

Herausgeber:

VEB Robotron-Projekt Dresden
Leningrader Straße 9
Dresden
DDR - 8010

ÁG 706/184/87 III/21/19

Inhaltsverzeichnis

1. Notation und Abkürzungen....7

2. FOR77 im Betriebssystem SCP....8

3. Übersetzen und Assemblieren von FORTRAN77-Programmen....9
3.1. Struktur des FOR77-Compilers....9
3.2. Aufruf des FOR77-Compilers....10
3.3. Dateien des FOR77-Compilers....11
3.3.1. Eingabedatei des FOR77-Compilers....11
3.3.2. Ausgabedateien des FOR77-Compilers....11
3.4. Optionen des FOR77-Compilers....12
3.4.1. Optionen in der Kommandozeile....15
3.4.2. Optionen der Steuerzeile....16
3.5. Auflistungen des FOR77-Compilers....18
3.5.1. Auflistungen auf dem Terminal....18
3.5.2. Auflistungen in der PRN-Datei....19
3.6. Aufbau des erzeugten Assemblerprogramms....20
3.7. Aufruf des Assemblers RASM86....22

4. Verbinden von FORTRAN77-Programmen....23
4.1. Aufruf des Linkers LINK86....24
4.2. Benutzung von Bibliotheken....24
4.3. Stack-Verkürzung bei der Programmverbindung....25
4.4. Anschluß von Assembler-Programmen....25

5. Ausführen von FORTRAN77-Programmen....27
5.1. Aufruf eines FORTRAN77-Programms....27
5.2. Standard-Vorverbindungen von Dateien....28

6. FORTRAN77-Ein/Ausgabe im Betriebssystem SCP....29
6.1. Dateistruktur und Dateizugriff....29
6.2. Implementierungsbedingte Besonderheiten der FORTRAN77-E/A....31

7. Gleitkomma-Arithmetik in FORTRAN77....33
7.1. Gleitkomma-Emulator....33
7.2. Ausnahmebedingungen des Gleitkomma-Emulators....36

8. Fehlermeldungen....40
8.1. Fehlermeldungen zur Übersetzungszeit....40
8.2. Fehlermeldungen zur Ausführungszeit....42
8.2.1. E/A-Fehlermeldungen....43
8.2.2. FORTRAN77-E/A-Fehlermeldungen....43
8.2.3. Festkomma- und Überlauffehler....44
8.2.4. Fehler bei der Abarbeitung von Gleitkomma-Funktionen....44
8.2.5. Gleitkomma-Fehler....44
8.2.6. Allgemeine Fehler....44
8.2.7. Stop durch Übersetzungszeitfehler....45

9. Optimierung von FORTRAN77-Programmen....46
9.1. Optimierungsstufe 1 (OPT=1)....46
9.2. Optimierungsstufe 2 (OPT=2)....47
9.3. Allgemeine Bemerkungen zur effektiven Programmierung....47

Anlage 1 Bereichsgrenzen und interne Darstellung der FORTRAN77-Datentypen....48
Anlage 2 Fehlernachrichten zur Übersetzungszeit....49

Anlage 3 Fehlercodes der Ausführungszeitfehler.....64
Anlage 4 Modul- und PUBLIC-Namen der Laufzeitbibliotheken des FOR77.....66
Anlage 5 Implementierungsbedingte Besonderheiten.....72

Literaturverzeichnis.....73

Sachwortverzeichnis.....74

Tabellenverzeichnis

Tabelle 1 Pässe des FOR77-Compilers.....9
Tabelle 2 Optionen der Kommandozeile.....15
Tabelle 3 Optionen der Steuerzeile.....16
Tabelle 4 Laufzeit-Initialisierungsroutinen.....21
Tabelle 5 Bibliotheken zur Programmverbindung.....23
Tabelle 6 Dateizugriffsarten.....30
Tabelle 7 IOSTAT-Werte.....32

Bildverzeichnis

Bild 1 STACK-Größe in FORTRAN77-Programmen.....21

1. Notation und Abkürzungen

Zur übersichtlichen Darstellung von einzugebenden Kommandozeilen oder von Ausschriften der Komponenten des SCP auf dem Terminal werden in vorliegender Schrift folgende Notationen und Abkürzungen verwendet.

Großbuchstaben	- für Kommandos
Kleinbuchstabe(n)	- für Ausschriften
'Kleinbuchstabe(n)'	- variable Teile in Ausschriften und Kommandozeilen
[ ]	- Kennzeichnung von wahlfreien Angaben in Kommandozeilen und Ausschriften; ([und] werden zur Darstellung der Symbole [und] verwendet)
!	- Formulierung von Alternativen (z.B. a!b)
{ }	- Klammerung
::=	- Definition und Untersetzung von variablen Teilen in Ausschriften und Kommandozeilen
CR	- Wagenrücklauf (Eingabetaste, hexadezimal 0D)
LF	- Zeilenvorschub (hexadezimal 0A)
CTRL-?	- gleichzeitiges Betätigen der Tasten CTRL und der anstelle des Zeichens '?' aufgeführten Taste.
FF	- Vorschub auf Anfang der nächsten Seite (hexadezimal 0C)

2. FOR77 im Betriebssystem SCP

FOR77 besteht aus dem Compiler und vier Laufzeitbibliotheken. Der FOR77-Compiler steht dem Anwender entweder auf drei 5 1/4"-Disketten oder auf drei 8"-Disketten zur Verfügung. Die Laufzeitbibliotheken einschließlich Gleitkomma-Emulator befinden sich auf einer 5 1/4"-Diskette oder auf einer 8"-Diskette.

Zur Übersetzung eines FORTRAN77-Programms mit dem FOR77-Compiler wird ein Hauptspeicherbereich von mindestens 200K benötigt. Größere Hauptspeicherkonfigurationen verringern die Übersetzungszeit.

Neben den Bestandteilen von FOR77 werden zur Herstellung eines ausführbaren FORTRAN77-Programms noch andere Komponenten des Betriebssystems SCP benötigt. Das betrifft im einzelnen den Assembler RASM86 und den Linker LINK86. In besonderen Fällen kann es auch sinnvoll sein, den Bibliothekar LIB86 zu verwenden. Zur Erzeugung und Wartung von Quelltexten und Objektmoduln werden die Serviceprogramme des SCP benötigt.

Mit Hilfe des FOR77-Compilers erfolgt die Übersetzung eines FORTRAN77-Programms in ein Assemblerprogramm. Das Assemblerprogramm muß mit dem Assembler RASM86 in einen Objektmodul umgewandelt werden. Durch eine anschließende Bearbeitung mit dem Linker LINK86 werden alle Bezugnahmen aus dem Objektmodul auf die Laufzeitbibliotheken aufgelöst und als Resultat entsteht das ausführbare FORTRAN77-Programm.

Ein einfaches Beispiel soll die Arbeit mit dem Sprachübersetzer FOR77 einleitend veranschaulichen. Folgende Kommandofolge ist denkbar:

```
A>FOR77   C:BEISP1   %ABEB CR
A>RASM86  B:BEISP1   %ABOB CR
A>LINK86  C:TEST=B:BEISP1.OBJ,F77INI.L86[S],F77EAS.L86[S],87NULL.
L86[S] CR
A>C:TEST (UNIT002=CON:) CR
```

3. Übersetzen und Assemblieren von FORTRAN77-Programmen

Bei der Übersetzung eines FORTRAN77-Programms wird ein Assemblerprogramm mit dem Dateityp A86 erzeugt. Das hat den Vorteil, daß der Anwender auf dem Assembler-Niveau Systemmittel des SCP einfügen kann. Beispielsweise seien in diesem Zusammenhang die DEBUG-Möglichkeiten, die Überlagerungsmöglichkeiten zur Ausführungszeit und die STACK-Verkürzungen für FORTRAN77-Programmeinheiten (siehe Abschnitt 3.6.) genannt.

3.1. Struktur des FOR77-Compilers

Der FOR77-Compiler übersetzt einen FORTRAN77-Quelltext in mehreren Schritten in ein Assemblerprogramm. Diese Übersetzungsschritte werden im weiteren Pässe genannt. Ein Paß kann aus mehreren Phasen bestehen, die nacheinander geladen werden. In Tabelle 1 sind die Pässe und ihre Funktionen dargestellt.

Tabelle 1 Pässe des FOR77-Compilers

Pass	Phase	Funktion
FIPASS	FIPASS	Compilerinitialisierung
WPASS		Lexikalische Analyse
	W1PASS	- der nichtausführbaren FORTRAN77-Anweisungen
	W2PASS	- der ausführbaren FORTRAN77-Anweisungen
SPASS		Syntaktische Analyse
	S1PASS	- der nichtausführbaren FORTRAN77-Anweisungen
	S2PASS	- der ausführbaren FORTRAN77-Anweisungen
	SDPASS	- Verarbeitung der DATA-Anweisungen
GXPASS	GXPASS	Globale Prüfung und Erzeugung von Symboltabellen und Symbolnachweistabellen, sowie Vorbereitung der Optimierung
OPASS	OPASS	Optimierung 1
DPASS	DPASS	Optimierung 2
RPASS	RPASS	Registerzuordnung
CPASS	CPASS	Codeerzeugung
EPASS	EPASS	Erzeugung der Fehlermeldungsliste

Der FOR77-Compiler benötigt für seine eigene Initialisierung die Datei FOR77.INI. Um diese Datei laden zu können, muß sie sich auf der Diskette befinden, auf der die Datei FIPASS.CMD liegt, oder das zugehörige Gerät muß in der I-Option (siehe Abschnitt 3.4.1) angegeben werden.

Der Compiler ist als Überlagerungsstruktur aufgebaut, in der die einzelnen Phasen als Überlagerungssegmente nachgeladen werden. Die Phasen sind teilweise wiederum Überlagerungsstrukturen. Es muß gesichert werden, daß eine Phase vollständig auf einer Diskette enthalten ist. Befindet sich eine Phase nicht auf der Diskette, von der die vorhergehende Phase geladen oder von der FOR77 aufgerufen wurde (siehe Abschnitt 3.2.), so erscheint auf dem Terminal die Meldung:

```
unable to open file: 'phase'
change disk and/or type device name
```

Dabei ist 'phase' eine der Phasen aus Tabelle 1.

Der Anwender hat an dieser Stelle die Möglichkeit, eine Diskette zu wechseln und die Gerätespezifikation (A-P) der gewechselten Diskette anzugeben. Handelt es sich um das Gerät, von dem die vorhergehenden Phasen geladen wurden, genügt CR. Liegt die neue Diskette auf einem anderen Gerät, muß die Gerätespezifikation eingegeben werden. Aus Gründen der Sicherheit wird eine Bestätigung der Gerätespezifikation durch die Aufforderung

```
Confirm device 'geräteangabe' :
```

verlangt. Die Aufforderung muß durch eine neue Geräteangabe oder mit CR beantwortet werden. Die Eingabe von CTRL-C beendet die Übersetzung.

Auf den Disketten, die während einer Übersetzung gewechselt werden, darf weder die Arbeitsdatei des Compilers noch die Quelltexteingabedatei angelegt sein.

3.2. Aufruf des FOR77-Compilers

Der FOR77-Compiler wird mittels folgendem Kommando aufgerufen:

```
FOR77 'qdatei'[{o!#}'options'] CR
```

Dabei ist

'qdatei' die Dateispezifikation des zu übersetzenden FORTRAN77-Quelltextes und

'options' eine Liste von Optionen der Kommandozeile (siehe Tabelle 2), mit deren Hilfe der Übersetzungsablauf und Compilerlisten/-dateien gesteuert werden.

Eine Dateispezifikation hat das im SCP gebräuchliche Format:

```
[geräteangabe:] dateiname [.dateityp]
```

Fehlt der Dateityp, so ergänzt der FOR77-Compiler den Dateityp F77.

Bei fehlender Geräteangabe wird das Gerät aus der S-Option (siehe Abschnitt 3.2.1.) ergänzt oder das Standardgerät verwendet.

Kommt es durch fehlerhafte oder fehlende Angaben in der Kommandozeile zu Initialisierungsfehlern des FOR77-Compilers, so wird die Übersetzung mit einer selbsterklärenden Meldung abgebrochen.

3.3. Dateien des FOR77-Compilers

3.3.1. Eingabedatei des FOR77-Compilers

Die Eingabedatei muß sich auf einer Diskette befinden. Ihre Dateispezifikation ist in der Kommandozeile anzugeben.
Um den Anwender die Eingabe im FORTRAN77-Standardeingabeformat (siehe [1]) zu erleichtern, wurde die Möglichkeit der Formatierung mittels Tabulator eingeführt. Ein Tabulator in den Spalten 1 bis 6 bewirkt den Übergang zu Spalte 6. Eine Ziffer nach dem Tabulator wird als Fortsetzungskennzeichnung gedeutet. Jedes andere Zeichen wird als erstes Zeichen des Anweisungsfeldes interpretiert. Die Zeichen vor dem Tabulator werden dem Markenfeld zugeordnet. Im Feld für Fortsetzungskennzeichnung und im Anweisungsfeld hat der Tabulator die gleiche Bedeutung wie ein Leerzeichen.
Tritt im Anweisungsfeld des Standardeingabeformates das Zeichen "!" auf, so wird der Rest der Zeile als Kommentar gewertet. Damit wurde in Erweiterung des Standards die übliche Art der Kommentierung ermöglicht.
Neben dem Standardeingabeformat erlaubt der FOR77-Compiler ein freies Format, das über die FREEFORM-Option (siehe Abschnitt 3.4.) ausgewählt werden kann.

3.3.2. Ausgabedateien des FOR77-Compilers

Der FOR77-Compiler erzeugt in Abhängigkeit von Optionen (siehe Abschnitt 3.2.1.) verschiedene Dateien. Diese Dateien erhalten eine Dateispezifikation gemäß Betriebssystemkonvention (siehe Abschnitt 3.2.).
Die Geräteangabe wird entweder einer Option der Kommandozeile entnommen oder es wird, falls die Geräteangabe fehlt, das Gerät der Eingabedatei angenommen. Der Dateiname ist immer der Dateiname des FORTRAN77-Quelltextes, der augenblicklich übersetzt wird.
Der Dateityp der erzeugten Datei wird wie folgt gebildet:

PRN - für Quelltextprotokoll, Symbol- und Symbolnachweistabelle

A86 - für den Assemblerquelltext, der als Resultat der Übersetzung entsteht

ERR - Fehlerprotokoll der Übersetzungszeitfehler

Der FOR77-Compiler benötigt eine Arbeitsdatei. Sie hat immer den Dateinamen SPILLFIL und den Dateityp TMP. Die Geräteangabe wird aus der T-Option (siehe Abschnitt 3.4.1.1.) entnommen oder es wird das Gerät der Eingabedatei benutzt. Diese Arbeitsdatei existiert nur temporär während der Übersetzungszeit.

3.4. Optionen des FOR77-Compilers

Die Optionen des FOR77-Compilers dienen der Steuerung des Übersetzungsprozesses und der Definition der Ein- und Ausgabedateien des Compilers.
Einige der Optionen sind nur in der Kommandozeile angebbar, andere nur im Quellprogramm (siehe Abschnitte 3.4.1. und 3.4.2.). Im Folgenden sind alle Optionen unabhängig von ihrer Position beschrieben. Die Namen der Optionen dürfen nur in gekürzter Form (siehe Tabellen 2 und 3) verwendet werden.

GOSTATEMENT/NOGOSTATEMENT

In der A86-Datei werden bei Angabe von GOSTATEMENT Anweisungen erzeugt, die die Anweisungsnummer in einen Arbeitsspeicherplatz eintragen.
Bei Fehlern zur Ausführungszeit wird dann die Anweisungsnummer mit ausgegeben.

DEBUG/NODEBUG

Ein in Spalte 1 eines FORTRAN77-Quelltextsatzes stehendes Zeichen "D" wird durch Leerzeichen ersetzt, wenn DEBUG wirksam ist und als FORTRAN-Anweisung übersetzt. Ist NODEBUG wirksam, wird eine solche Zeile als Kommentarzeile aufgefaßt. DEBUG impliziert GOSTATEMENT.

ALIST/NOALIST

Im Assemblertext der A86-Datei wird an dieser Stelle eine LIST-Direktive erzeugt, wenn ALIST angegeben wird. NOALIST führt zu einer NOLIST-Direktive.

SYMBOLS/NOSYMBOLS

In der PRN-Datei wird eine Symboltabelle erzeugt, wenn SYMBOLS angegeben wurde.
Die Tabelle enthält die Namen und Attribute der im Programm vereinbarten Objekte.

XREF/NOXREF

In der PRN-Datei wird eine Symbolnachweistabelle erzeugt, wenn XREF angegeben wurde. Ist XREF und SYMBOLS wirksam, entsteht eine gemeinsame Tabelle.

PAGELENGTH(n)

Bei der Erzeugung der PRN-Datei wird nach n Zeilen ein Seitenvorschub erzeugt.
Dabei gilt $1 < n < 32760$.

PAGEWIDTH(n)

Die Zeilenlänge einer Druckzeile der PRN-Datei beträgt maximal n Zeichen.
Dabei gilt $60 < n < 132$.

ERRORLIMIT(n)

Nach n Fehlermeldungen erfolgt der Abbruch der Übersetzung. Dabei gilt n>=0, wobei n=0 bedeutet, daß kein Abbruch der Übersetzung auf Grund einer Fehleranzahl erfolgen soll.

STORAGE(INTEGER*X,REAL*Y,LOGICAL*Z)

Mit dieser Option wird die Länge von INTEGER-, REAL- und/oder LOGICAL-Variablen in Bytes festgelegt, wenn sie im FORTRAN77-Programm ohne Längenangabe vereinbart werden. Für x, y und z sind folgende Werte möglich:

x = 2 oder 4
y = 4 oder 8
z = 1, 2 oder 4

Wird die STORAGE-Option nicht angegeben, gelten folgende Standardannahmen:

x = 4
y = 4
z = 4

COMPILE

Ist diese Option wirksam, wird die Übersetzung über den SPASS (siehe Abschnitt 3.1.) hinaus fortgesetzt, auch wenn Fehler der Schwere S (siehe Abschnitt 8.1.) im FORTRAN77-Quellprogramm auftraten.

NOCOMPILE[(n)]

Ist die Option NOCOMPILE wirksam, wird die Übersetzung nach dem GXPASS unbedingt beendet. Durch die Angabe von NOCOMPILE(n) ist die Fortsetzung der Übersetzung abhängig von der Fehlerschwere n. Wenn Fehler der Schwere n auftraten, wird die Übersetzung nach dem GXPASS abgebrochen. Dabei kann n das Zeichen E oder S sein.

TITLE('text')

Die Zeichenkette 'text' wird in die erste Zeile einer Seitenüberschrift in der PRN-Datei eingefügt.

SUBTITLE('text')

Die Zeichenkette 'text' wird in eine zusätzlich angegebene zweite Überschriftszeile der PRN-Datei eingefügt. Tritt eine Steuerzeile mit SUBTITLE-Option zwischen FORTRAN77-Anweisungen auf, bewirkt sie gleichzeitig den Übergang auf eine neue Seite der PRN-Datei.

EJECT

Übergang auf eine neue Seite der PRN-Datei.

QUIET

Fehlermeldungen zur Übersetzungszeit werden nicht auf dem Terminal ausgegeben, sie werden nur in der ERR-Datei gesammelt. Ist QUIET nicht wirksam, wird jede Fehlermeldung bei Erkennen des Fehlers auf dem Terminal in einer Kurzform ausgegeben (siehe Abschnitt 8.1.).

NOOPT

Es wird keine Optimierung (KPASS, OPASS, DPASS) durchgeführt.

OPT[(n)]

Optimierung des FORTRAN77-Programms. Dabei gilt:

n = 1 einfache Optimierung auf lokalem Niveau
n = 2 erweiterte Optimierung auf globalem Niveau

Wird (n) nicht angegeben, gilt OPT(1).

VALUE-CONTROL/NOVALUE-CONTROL

Es wird Code für die Überwachung von Werten der Indexgrößen und von Längenwerten erzeugt oder nicht erzeugt.

FREEFORM/NOFREEFORM

Bei Angabe der Option FREEFORM kann der Eingabetext im freien Format eingegeben werden. Das Zeichen '@' als letztes Zeichen einer Zeile zeigt in diesem Fall an, daß die nächste Zeile als Fortsetzungszeile gedeutet werden soll. Marken und Anweisungen können bei freiem Format an beliebigen Zeilenpositionen beginnen.
Tritt das Zeichen '@' innerhalb einer Zeichenkette auf, die über mehrere Zeilen geht, und ist das letzte Zeichen auf einer Zeile das Zeichen '@', so ist dafür '@@' eizugeben.
Ist das erste Zeichen einer Zeile (Spalte 1) eines der Zeichen 'C', '*' oder Leerzeichen, so wird die Zeile als Kommentarzeile gedeutet. Kommentarzeilen innerhalb von Anweisungen sind nicht gestattet.
Mit Angabe der NOFREEFORM-Option wird das Standardformat für den Eingabetext gefordert.

INCLUDE(datei_name)

Die INCLUDE-Option liefert die Möglichkeit, Quelltext aus einer weiteren Datei in das Programm einzufügen. Im eingefügten Quelltext können wiederum INCLUDE-Steuerzeilen enthalten sein. Die Schachtelungstiefe ist auf drei Stufen beschränkt.
Ist "dateiname" in der INCLUDE-Option nicht vollständig angegeben, d.h. "gerät: dateiname.erweiterung", so werden folgende Standardannahmen gültig:
"gerät" : Geräteangabe aus der S-Option der Kommandozeile, bzw. Gerät der primären Eingabe
"erweiterung" : F77

3.4.1. Optionen in der Kommandozeile

Die Optionen der Kommandozeile beginnen mit dem Zeichen / oder # und müssen vom Dateinamen der Quelltextdatei durch mindestens ein Leerzeichen getrennt sein. Die Optionen untereinander können durch Leerzeichen getrennt sein. Wenn eine Option aus zwei Zeichen besteht, dürfen diese nicht durch Leerzeichen getrennt sein. Die Tabelle 2 enthält eine Übersicht aller Optionen der Kommandozeile.

Tabelle 2 Optionen der Kommandozeile (Teil 1)

Angabe der Option in der Kommandozeile		Standard, falls nicht angegeben	Bedeutung
1. Z.	2. Zeichen		
A		Gerät der Quelltext-eingabe	Gerät für A86-Datei
	Z		keine A86-Datei
	X 3)		Ausgabe auf Terminal
	Y		Ausgabe auf Drucker
	/		Ausgabe auf Standardgerät
	A bis P		Ausgabe auf Gerät A bis P
P	wie Option A	Z	Gerät für PRN-Datei (siehe Abschnitt 3.2.2.)
T	/	Gerät der Quelltext-eingabe	Gerät für Arbeitsdatei SPILLFIL.TMP
	A bis P		
S	/	Geräteangabe der Quelltext-datei oder /	Gerät für Quelltext-Eingabedatei
	A bis P		
E	wie Option A	Gerät der Quelltext-eingabe	Gerät für ERR-Datei (siehe Abschnitt 3.2.2.)
D	-	NODEBUG	DEBUG 1)
I	/	Gerät, auf dem sich FIPASS.CMD befindet	Gerät, auf dem sich die Diskette mit der Datei FOR77.INI befindet (siehe Abschnitt 3.1.)
	A bis P		
G	-	NOGOSTATEMENT	GOSTATEMENT 1)
F	S oder Leerz.	NOFREEFORM	NOFREEFORM 1)
	F		FREEFORM
V	-	NOVALUE-CONTROL	VALUE-CONTROL 1)

Tabelle 2 Optionen der Kommandozeile (Teil 2)

Angabe der Option in der Kommandozeile		Standard, falls nicht angegeben	Bedeutung	
1. Z.	2. Zeichen			
B	-	NOALIST	ALIST	1)
Q	-	-	QUIET	2)
C	-	NOCOMPILE(S)	COMPILE	1)
@	1 oder Leerz.	NOOPTIMIZE	OPT(1)	1)
	2		OPT(2)	
N	C	NOCOMPILE(S)	NOCOMPILE	1)
	E		NOCOMPILE(E)	
	L	LIST	NOLIST	1)
X	-	NOXREF NOSYMBOLS	XREF, SYMBOLS	1)

Erläuterungen:

1) Die Beschreibung der angegebenen Optionen ist Abschnitt 3.4. zu entnehmen.

2) Ist die Option QUIET wirksam, wird keine Meldung auf dem Terminal ausgegeben, wenn ein Fehler im FORTRAN77-Quellprogramm gefunden wird (siehe Abschnitt 8.1.)

3) Geräteangabe:

A bis P	Gerät 'A:' bis 'P:'
X	CON:
Y	LST:
Z	NUL: dient der Unterdrückung der Ausgabe
@	Standardgerät

3.4.2. Optionen der Steuerzeile

Steuerzeilen eines FORTRAN77-Programms sind durch das Zeichen o als erstes Zeichen der Zeile gekennzeichnet. Beginnt eine Steuerzeile mit der Zeichenfolge oo, wird die Steuerzeile nicht auf die Listendatei (Dateityp PRN) übertragen.
Diesen ersten Zeichen folgt eine durch Kommas getrennte Liste von Optionen, wie sie laut Tabelle 3 in einer Steuerzeile angegeben werden dürfen.

Tabelle 3 Optionen der Steuerzeile

Angabe der Option in der Steuerzeile	Option 1)	Position im FORTRAN77-Quelltext 2)
[NO]SB	[NO]SYMBOL	P
[NO]GS	[NO]GOSTATEMENT	P
[NO]L	[NO]LIST	A
[NO]XR	[NO]XREF	P
EJ	EJECT	A 3)
PW(n)	PAGEWIDTH	P
PL(n)	PAGELENGTH	P
OPT[(n)]	OPTIMIZE	P
NOOPT	NOOPTIMIZE	P
[NO]AL	ALIST	A
STG([I*x,][R*y,][L*z]) 4)	STORAGE	P
FF	FREEFORM	P
[NO]DB	[NO]DEBUG	P
[NO]VC	[NO]VALUE-CONTROL	P
STT('kette')	SUBTITLE	A
TT('kette')	TITLE	P
COM	COMPILE	P
NOCOM[(n)]	NOCOMPILE	P
EL(n)	ERRORLIMIT	P
NOEL	NOERRORLIMIT	P
INC(datei_name)	INCLUDE	A

Erläuterungen:

1) siehe Abschnitt 3.4.

2) P - vor der ersten FORTRAN77-Quelltextzeile
A - vor oder zwischen FORTRAN77-Quelltextzeilen

3) nicht vor der ersten FORTRAN77-Quelltextzeile

4) x, y, z siehe Abschnitt 3.4.

3.5. Auflistungen des FOR77-Compilers

3.5.1. Auflistungen auf dem Terminal

Zu Beginn der Übersetzung gibt der Compiler folgende Informationen auf dem Terminal aus:

- Kopfzeile
  FORTRAN77 Compiler 1700(SCPX) V.M n.m

- Dateispezifikation der Quelltextdatei, wie sie in der Kommandozeile angegeben wurde und die in der Kommandozeile angegebenen Optionen

- Dateispezifikationen aller vom Compiler benutzten Dateien in folgender Form:

  File assignments:

  Source-file: 'Dateispezifikation'
  Print-file: 'Dateispezifikation'
  Spill-file: 'Dateispezifikation'
  A86-file: 'Dateispezifikation'
  ERR-file: 'Dateispezifikation'

Wird während der Übersetzung eine weitere Phase des Compilers geladen, so erscheint auf dem Terminal die Ausschrift:

```
Phase: 'Name der Phase'
(z.B.  Phase: W1PASS)
```

Wird die zu ladende Phase auf der Diskette g, von der die vorherige Phase geladen wurde, nicht gefunden, so erscheinen auf dem Terminal folgende Mitteilung und Aufforderung:

```
Unable to open g: 'name der Phase'
Change disk and <CR> or type device name
```

Auf diese Aufforderung ist die Diskette mit der zu ladenden Phase einzulegen und/oder der Name des Laufwerkes (A-P) einzugeben, das die Diskette mit der zu ladenden Phase enthält. Wird das Laufwerk nicht gewechselt, so reicht nach Einlegen der neuen Diskette das Drücken der Eingabetaste.
Mittels CTRL-C kann aber auch zu einem solchen Zeitpunkt die Übersetzung abgebrochen werden.
Wird die obige Aufforderung mit der Angabe des Namens eines Laufwerks befolgt, so erscheint auf dem Terminal die Aufforderung

```
Confirm device 'gerätename': >
```

Mit dem Drücken der Eingabetaste wird bestätigt, daß das richtige Laufwerk angesprochen wurde, bzw. eine neue Geräteangabe wird eingegeben.

Zu Beginn der Übersetzung wird nach der Verarbeitung der führenden Steuerzeilen innerhalb des Quelltextes (bis zur ersten FORTRAN77-Anweisung) eine Liste der wirksamen Optionen auf dem Terminal ausgegeben.

Weitere Auflistungen auf dem Terminal erscheinen, wenn die PRN-Datei auf das Terminal gelegt wurde (siehe Abschnitt 3.5.2.) oder wenn die Option QUIET nicht angegeben wurde.
Wurde die Option QUIET nicht angegeben, so wird bei jedem Fehler im FORTRAN77-Programm eine verkürzte Fehlermeldung auf das Terminal ausgegeben. Dabei besteht die Möglichkeit, die Übersetzung abzubrechen (siehe Abschnitt 8.1.).

3.5.2. Auflistungen in der PRN-Datei

Wird in der Kommandozeile die P-Option mit einem von Z verschiedenen Gerätenamen angegeben, so wird die PRN-Datei auf dem entsprechenden Gerät erzeugt.
Die erste Zeile der Auflistungen in der PRN-Datei enthält Version und Modifikation des Compilers, die Dateispezifikation des übersetzten Moduls und die Seitennummer. Diese Zeile entfällt, wenn die Ausgabe auf Terminal erfolgt.

```
FORTRAN-77 COMPILER 1700(SCPX)  V.M 'n.m' MODULE: dat_spez  PAGE 1
```

Die folgenden Zeilen enthalten eine Auflistung der Steuerzeilen bis zur ersten FORTRAN77-Anweisung und die Auflistung der für die Übersetzung wirksamen Optionen.

Ist die Option LIST wirksam, so wird anschließend der FORTRAN77-Quelltext aufgelistet.
Die Quelltextauflistung hat folgenden Aufbau:

1. Kopfzeile

```
FOR77  n.m     MODULE: dateispezifikation          PAGE: 'i'
                wenn keine TITLE-Option vorhanden

FOR77  n.m      'Zeichenkette aus TITLE-Option'   PAGE: 'i'
                wenn TITLE-Option vorhanden
```

2. Kopfzeile
   Sie ist nur vorhanden, wenn die SUBTITLE-Option angegeben wurde und enthält die Zeichenkette aus der SUBTITLE-Option.

3. Kopfzeile

```
LINE STMT        S O U R C E   L I S T I N G
```

In den folgenden Zeilen wird jede Quelltextzeile mit Zeilen- und Anweisungsnummer ausgegeben. Überschreitet die Länge der Quelltextzeile den Wert in der PAGEWIDTH-Option, so wird eine weitere Zeile mit dem Rest der Quelltextzeile ausgegeben.

Die Optionen EJECT und SUBTITLE bewirken den Übergang auf eine neue Seite.

Sind die Optionen XREF und/oder SYMBOLS wirksam, so wird eine Symbol- und/oder Symbolnachweis-Tabelle ausgegeben.

Die Tabelle besitzt folgenden Aufbau:

Spalten 1 bis 10 Name des im Programm vereinbarten Objekts

Spalten 11 bis n (n=Angabe in PAGEWIDTH)
falls die Option
- SYMBOL angegeben wurde, wird eine Zeile mit den Attributen des vereinbarten Objekts ausgegeben.
- XREF angegeben wurde, wird eine oder mehrere Zeilen mit den Referenzen des vereinbarten Objekts ausgegeben.
Das einer Referenz beigefügte Zeichen "*" zeigt an, daß das betreffende Objekt in der Anweisung deren Nummer als Referenz angegeben ist, als Marke vereinbart wurde oder eine Wertänderung erfahren kann.

3.6. Aufbau des erzeugten Assemblerprogramms

Der FOR77-Compiler erzeugt für eine FORTRAN77-Programmeinheit in der Datei mit dem Dateityp A86 ein Assemblerprogramm mit folgendem Aufbau:

NAME-Direktive — mit dem Namen der FORTRAN77-Programmeinheit.

EXTRN-Direktive(n) — für im Programm vereinbarte oder compilerintern erzeugte externe Namen.

Extra-Segment(e) — für im Programm vereinbarte COMMON-Bereiche. Die einleitende ESEG-Direktive hat folgenden Aufbau:

{'COMMON-Name' ! @COMMON} ESEG PARA COMMON

Für den unbenannten COMMON-Bereich lautet der ESEG-Name aCOMMON.

Daten-Segment — für Programmvariablen, Feldbeschreiber, Formatlistenbeschreiber und Konstante
Das Segment wird der Speicherklasse 'CODE' zugeordnet. Die Segment-Direktive hat folgenden Aufbau:

D@XXX DSEG PARA PUBLIC 'CODE'

XXX - Name der Programmeinheit

Code-Segment mit den Assemblerbefehlen zur Realisierung des FORTRAN77-Programms, bestehend aus:
- Code für FORTRAN77-Anweisungen, jeweils durch Kommentar eingeleitet
- Code für compilerinterne Unterprogramme
- LIST/NOLIST-Direktiven (siehe Abschnitt 3.4.)

Die Segment-Direktive hat folgenden Aufbau:

C@XXX CSEG PUBLIC

XXX - Name der Programmeinheit

Stack-Segment für Rettebereiche, Parameterübergaben usw.

Das Code-Segment eines FORTRAN77-Hauptprogramms beginnt immer mit dem Aufruf der Laufzeit-Initialisierungsroutine FQ_PGM. Diese ruft weitere Initialisierungsroutinen auf (s. Tabelle 4).

Tabelle 4 Laufzeit-Initialisierungsroutinen

Name	Funktionen
F77_INI	Initialisierung des E/A-Steuersystems Initialisierung des Fehlerbehandlungssystems
INITFP	Initialisierungen für Gleitkomma-Arithmetik

Der Name INITFP ist durch die Gleitkomma-Bibliothek des Betriebssystems vorgegeben. Dieser Name ist somit nicht als Name einer externen Größe in irgendeiner FORTRAN77-Programmeinheit verwendbar. Das Stack-Segment einer FORTRAN77-Programmeinheit hat genau die Größe, die zur Ausführung dieser Programmeinheit notwendig ist. Während des Verbindens eines FORTRAN77-Programms (siehe Abschnitt 4.) ergibt sich die Länge des Stack-Segments aus der Summe der Längen aller Stack-Segmente der zu verbindenden FORTRAN77-Programmeinheiten.

Diese Stack-Länge ist aber zu groß, wenn sich aus der FORTRAN77-Programmstruktur ergibt, daß sich die bestimmten Programmeinheiten zugeordneten Stacksegmente überlagern.

Bild 1 soll ein Beispiel einer FORTRAN77-Programmstruktur liefern. Bei der in diesem Bild gezeigten Programmstruktur ist nur eine Gesamtlänge des Stacksegments von 435 Bytes notwendig, was der Summe aus den Längen der Stacksegmente des Hauptprogramms, des Unterprogramms A und des Unterprogramms C entspricht. Die Verkürzung des Stacks auf diese Länge von 435 Bytes ist auf zwei Arten zu erreichen. Einmal ist es möglich, beim Verbinden eine Maximalgröße des Stacks anzugeben (siehe Abschnitt 4.). Zum anderen kann aber schon auf dem Niveau des Assemblerquelltextes eingegriffen werden, indem zum Beispiel in den FORTRAN77-Programmen B und D nach Bild 1 mittels Editor ED die Stacksegmente entfernt werden.

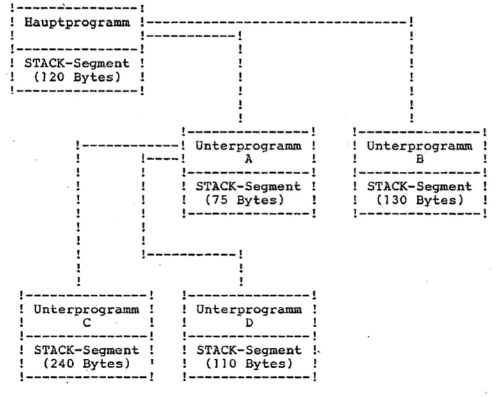


Bild 1 STACK-Größe in FORTRAN77-Programmen

3.7. Aufruf des Assemblers RASM86

Für die Assemblierung eines vom FOR77-Compiler erzeugten Assemblerprogramms ist die einfachste Form des Aufrufs des Assemblers RASM86 ausreichend. Dieser Aufruf hat folgende Form:

RASM86 'datname' CR

Dabei ist

'datname' die Dateispezifikation des Assemblerprogramms, das von FOR77 erzeugt wurde. Die Geräteangabe ist nur erforderlich, wenn diese Datei nicht auf dem Standardgerät liegt. Der Dateityp wird von FOR77 RASM86-gerecht erzeugt, muß also nicht angegeben werden.

Weitere Angaben sind nicht notwendig, um einen Objektmodul eines FORTRAN77-Programms erzeugen zu können.
Dem Anwender steht es natürlich frei, andere Möglichkeiten des Assemblers RASM86 zu benutzen. Eine exakte Beschreibung dieser Möglichkeiten kann [2] und [3] entnommen werden.

Vor der Verarbeitung mit RASM86 kann der Anwender auch den von FOR77 erzeugten Assemblerquelltext modifizieren, will er zum Beispiel eine Überlagerungsstruktur mittels der FORTRAN77-Programmeinheiten realisieren oder Testhilfen einbauen.

4. Verbinden von FORTRAN77-Programmen

Nach dem Übersetzen und Assemblieren aller zu einem FORTRAN77-Programm gehörenden Programmeinheiten besteht der nächste Arbeitsschritt aus dem Verbinden der FORTRAN77-Programmeinheiten untereinander, dem Verbinden mit den Routinen der Laufzeitbibliotheken und falls erforderlich mit dem Gleitkomma-Emulator.
In den zu verbindenden FORTRAN77-Programmeinheiten muß genau ein FORTRAN77-Hauptprogramm enthalten sein. Zur Auflösung der externen Bezugnahmen aus den assemblierten FORTRAN77-Programmen stehen die Bibliotheken des Betriebssystems und die des FOR77 zur Verfügung. Tabelle 5 enthält eine Übersicht aller Bibliotheken, die zum Verbinden notwendig sind. Wenn Gleitkomma-Arithmetik verwendet wird, muß zusätzlich der Gleitkomma-Emulator des Betriebssystems eingebunden werden. Der Gleitkomma-Emulator hat die Dateispezifikation EWM87.OBJ.
Zum Verbinden von Programmen steht im SCP der Linker LINK86 zur Verfügung. Diese Komponente bietet eine Vielzahl von Möglichkeiten, die an dieser Stelle nicht beschrieben werden. Dazu muß auf [4] verwiesen werden. In diesem Abschnitt sollen nur die prinzipiellen Möglichkeiten gezeigt werden, die zum Aufbau eines lauffähigen FORTRAN77-Programms notwendig sind.

Tabelle 5 Bibliotheken zur Programmverbindung

Dateiname	Bestand-teil von	Inhalt und Bedeutung
F77INI.L86	FOR77	Initialisierung und Programmbeendigung; wird stets benötigt
EWM87.L86	SCP	Gleitkomma-Initialisierungsroutinen; 1) wird benötigt, wenn Gleitkomma-Operationen oder Standardfunktionen mit Gleitkomma-Datentyp verwendet werden
87NULL.L86	SCP	Pseudoroutinen; 1) wird angegeben, wenn EWM87.L86 nicht benötigt wird
F77ART.L86	FOR77	Notwendig bei Verwendung von FORTRAN77-Standardroutinen, die Gleitkomma-Arithmetik benutzen
F77EAS.L86	FOR77	E/A-Routinen; wird nur benötigt, wenn E/A-Anweisungen oder Hilfs-E/A-Anweisungen verwendet werden.
EANULL.L86	FOR77	Pseudoroutinen; 1) wird angegeben, wenn F77EAS.L86 nicht benötigt wird.

1) Bei der Angabe der SEARCH-Option (siehe 4.2) ist zu beachten, daß diese Bibliothek im LINK-Kommando (siehe 4.1) hinter F77INI.L86 angegeben werden muß.

4.1. Aufruf des Linkers LINK86

Das Kommando zum Aufruf des Linkers hat folgende Form:

```
LINK86 ['datnam'=] 'datopt' [,'datopt']...
```

Dabei ist

'datnam' die Dateispezifikation des zu erzeugenden Programms.
Der Dateiname wird durch den Dateityp CMD ergänzt. Fehlt 'datnam'= , so wird der Name aus der ersten 'datopt'-Angabe zur Bildung des Programmnamens, ergänzt durch .CMD, verwendet.

'datopt' ::= 'datspez'[['option']]

'datspez' die Dateispezifikation eines Objektmoduls (Dateityp OBJ) oder einer Bibliothek (Dateityp L86).

'option' eine Liste von Optionen

Es soll noch auf die Möglichkeit hingewiesen werden, daß nach LINK86 eine Dateispezifikation und die Option INPUT angegeben werden kann (siehe [4]).

Die in den Linkprozess einzubeziehenden Dateien mit dem Dateityp OBJ können sowohl FORTRAN77-Programmeinheiten als auch in Assembler geschriebene Unterprogramme sein. Sobald in einer der zu verbindenden Programmeinheiten Gleitkomma-Operationen oder Gleitkomma-Funktionen ausgeführt werden, ist auch der Gleitkomma-Emulator EWM87.OBJ anzugeben.

4.2. Benutzung von Bibliotheken

Als Dateien mit dem Dateityp L86 können eigene mit dem Bibliothekar LIB86 (siehe [6]) erzeugte Bibliotheken angegeben werden. In solchen Bibliotheken können zum Beispiel aus FORTRAN77-Programmeinheiten erzeugte Objektmoduln zusammengefaßt sein, die vollständig oder auch teilweise für das zu verbindende FORTRAN77-Programm benötigt werden.
Aus Tabelle 5 ist ersichtlich, daß von den dort aufgeführten Laufzeitbibliotheken beim Verbinden immer mindestens drei angegeben sein müssen. Die Bibliothek F77INI.L86 wird immer benötigt. Die Bibliotheken F77ART.L86 und EWM87.L86 werden nur benötigt, wenn in den zu verbindenden FORTRAN77-Programmeinheiten Objekte mit REAL, DOUBLE PRECISION oder COMPLEX vereinbart wurden oder Standardfunktionen oder mathematische Operationen gemäß Tabelle 8 auftreten. Liegen diese Bedingungen nicht vor, so muß EWM87.L86 durch 87NULL.L86 ersetzt werden. Außerdem kann der Gleitkomma-Emulator entfallen.

Enthält ein zu verbindendes FORTRAN77-Programm auch nur eine E/A-oder eine Hilfs-E/A-Anweisung, muß im LINK86-Kommando die Bibliothek F77EAS.L86 angegeben werden. Ist im gesamten FORTRAN77-Programm keine dieser Anweisungen vorhanden, kann die Bibliothek F77EAS.L86 durch EANULL.L86 ersetzt werden. Das führt zu einer beträchtlichen Verringerung der Programmgröße.

An dieser Stelle ist der Hinweis auf eine der Optionen angebracht, die nach einer Dateispezifikation für eine Bibliothek angegeben werden kann. Es ist dies die Option SEARCH. Sie bewirkt, daß aus einer Bibliothek nur die Objektmoduln mit dem Programm verbunden werden, auf die auch Bezugnahmen bestehen. Fehlt diese Option, wird die gesamte Bibliothek in das Programm einbezogen.

Abschließend ein Beispiel für den Aufruf des Linkers:

```
A>B:LINK86
C:TEST.OBJ,F77INI.L86[S],F77ART.L86[S],F77EAS.L86[S],EWM87.L86[S],
EWM87.OBJ CR
```

In diesem Beispiel soll eine FORTRAN77-Programmeinheit, dargestellt durch den Objektmodul TEST.OBJ, zu dem Programm TEST.CMD mit allen benötigten Routinen der Laufzeitbibliotheken verbunden werden. Die FORTRAN77-Programmeinheit enthält Gleitkomma-Datentypen und E/A-Anweisungen.
Der Linker befindet sich auf dem Gerät B, die Laufzeitbibliotheken und der Gleitkomma-Emulator sind auf dem Standardgerät A untergebracht.

4.3. Stack-Verkürzung bei der Programmverbindung

Die Problematik der Stack-Größe eines FORTRAN77-Programms ist im Abschnitt 3.6. ausführlich erläutert worden. In diesem Abschnitt soll die alternative Lösung gezeigt werden, d.h. wie man zur real benötigten Stack-Größe gelangt. Der Anwender muß anhand der gegebenen Programmstruktur die erforderliche Größe des Stacks errechnen und diese Größe im MAX-Parameter des LINK86-Kommandos für den Stack angeben. Gemäß dem Beispiel nach Bild 1 hätte das LINK86-Kommando folgendes Aussehen:

```
A>B:LINK86 TEST[STACK[MAX[435]]]...BIBLIOTHEKEN ... CR
```

4.4. Anschluß von Assembler-Programmen

Aus FORTRAN77-Programmen werden während ihrer Übersetzung Assemblerprogramme erzeugt.
Es bietet sich deshalb für den Anwender an, eigene Assemblerprogramme zu schreiben und diese beim Verbinden in das FORTRAN77-Programm einzubinden.
Solche Assembler-Unterprogramme müssen den Konventionen der Assemblerprogramme entsprechen, die vom FOR77-Compiler erzeugt werden (siehe Abschnitt 3.6.).

An dieser Stelle sollen besonders die Anschlußbedingungen beim Betreten und Verlassen von Assembler-Unterprogrammen betrachtet werden.

Der Name eines externen Unterprogramms wird vom FOR77-Compiler wie folgt vereinbart:

```
EXTRN up-name FAR
```

Daraus folgt, daß ein eigenes Assembler-Unterprogramm stets mit RETF verlassen werden muß.
Beim Betreten eines Unterprogramms enthalten die Register SS und DI die Adresse der Argumentliste. Diese Liste hat folgenden Aufbau:

Offset	Länge	Inhalt
0	4	[A(Funktionswert)]
4	4	A(1.Argument)
.	.	.
.	.	.
.	.	.
x*4	4	A(x.Argument)

Am Offset 0 ist immer Platz für die Rückgabe eines Funktionswertes reserviert. Hat das Unterprogramm keinen Funktionswert, bleiben die ersten 4 Byte der Argumentliste unbenutzt. Ist ein Argument vom Datentyp CHARACTER, zeigt A(Argument) auf einen Beschreiber mit folgendem Aufbau:

Offset	Länge	Inhalt
0	4	A(Zeichenkettenargument)
4	2	Länge des Zeichenketten-Arguments

Vor dem Rücksprung in das aufrufende Programm müssen folgende Zustände hergestellt werden:

- Alle Register (außer DI) müssen den Inhalt zum Zeitpunkt des Aufrufes haben.
- Die Gleitkommaumgebung muß den Zustand zum Zeitpunkt des Aufrufs haben.
- Der Gleitkomma-Stapel muß leer sein.

5. Ausführen von FORTRAN77-Programmen

Ein FORTRAN77-Programm wird ausgeführt, wenn es unter dem vom Linker erzeugten Dateinamen aufgerufen wird. Dazu wird die Dateispezifikation des Programms als Kommando auf dem Terminal eingegeben.

5.1. Aufruf eines FORTRAN77-Programms

Zusätzlich können beim Aufruf Vorverbindungen zwischen Einheiten und FORTRAN77-Dateien hergestellt werden. Die vollständige Kommandozeile für den Aufruf ist:

```
'pname' ['datspez'!'liste']
```

Dabei ist

'pname' die Dateispezifikation des FORTRAN77-Programms. Der Dateityp .CMD muß nicht angegeben werden.

'datspez' die Dateispezifikation einer Datei. Diese Datei enthält in jeder Zeile eine Dateivorverbindung 'datver'.

'liste' eine Liste von Dateivorverbindungen 'datver' ('datver'[,'datver']...)

'datver' eine Dateivorverbindung der Form: UNIT 'datnr' = 'fspez'

'datnr' eine maximal dreistellige Einheitennummer (siehe [1], Abschnitt 6.1.2.)

'fspez' eine vollständige Dateispezifikation oder eine der folgenden Angaben:

Angabe	Bedeutung
X:	Terminal
Y:	Drucker
Z:	Pseudo-Datei
CON:	Terminal
LST:	Drucker
NUL:	Pseudo-Datei
RDR:	Lochbandleser
PUN:	Lochbandstanzer
WRK:	temporäre Datei auf Standardgerät
WKA:, WKB:,...WKE:	temporäre Datei auf Gerät A, B,..., E

Die Möglichkeit, die Dateivorverbindungen in einer Datei zeilenweise zu fixieren, besteht deshalb, weil pro Kommandozeile nur 128 Zeichen zur Verfügung stehen. Dateivorverbindungen, die sich in einer solchen Datei oder in der Kommandozeile befinden, können für eine Einheitennummer mehrfach auftreten. Es ist aber zu beachten, daß die erste aufgetretene Dateivorverbindung wirksam wird.

5.2. Standard-Vorverbindungen von Dateien

Auch wenn keine Vorverbindungen beim Programmaufruf angegeben werden, existieren zwei Standard-Vorverbindungen; eine für UNIT005 als Eingabedatei und eine für UNIT006 als Ausgabedatei. Diese Vorverbindungen haben die Form:

```
UNIT005=CON:
UNIT006=CON:
```

und sichern die Verwendbarkeit von PRINT-Anweisungen und von READ- und WRITE-Anweisungen, in denen der UNIT-Parameter nicht oder in denen UNIT=* angegeben wurde. Die Standard-Vorverbindungen können durch die Angabe eigener Vorverbindungen für UNIT005 und UNIT006 beim Programmaufruf überschrieben werden.

Als Beispiel soll ein FORTRAN77-Programm TSTBSP ausgeführt werden. Es benötigt Dateien mit den Einheitennummern 1 und 6:

```
A>B:TSTBSP (UNIT001=C:DAT1.TXT,UNIT006=LST:) CR
```

6. FORTRAN77-Ein/Ausgabe im Betriebssystem SCP

Im Betriebssystem SCP wird eine Datei durch eine Dateispezifikation (siehe Abschnitt 5.) beschrieben. Das Mittel der Sprache FORTRAN77 für den Zugriff auf eine Datei ist die Einheit. Für den Zugriff auf eine Datei muß eine aktive Verbindung zwischen dieser Datei und der zum Zugriff benutzten Einheit hergestellt werden. Diese Verbindung wird über die Einheitennummer und die Dateispezifikation erzeugt.
Aktive Dateiverbindungen entstehen
- durch OPEN-Anweisungen oder
- durch eine READ-, WRITE- oder PRINT-Anweisung, wenn für die angegebene Einheit eine Dateivorverbindung existiert.

Eine aktive Dateiverbindung muß eindeutig sein, d.h. es darf zur gleichen Zeit eine Einheit höchstens mit einer Datei aktiv verbunden sein. Eine aktive Dateiverbindung wird durch eine CLOSE-Anweisung oder am Programmende inaktiv.

Dateivorverbindungen können beim Aufruf des FORTRAN77-Programmes erzeugt werden (siehe Abschnitt 5.). Alle Dateivorverbindungen sind inaktiv. Sie erlauben die Benutzung von Einheiten, ohne daß zuvor für diese Einheiten eine OPEN-Anweisung ausgeführt wurde. Eine Dateivorverbindung für eine Einheit bleibt unberücksichtigt, falls in einer OPEN-Anweisung für diese Einheit der FILE-Parameter mit einer Dateispezifikation angegeben wird. Existiert für eine Einheit keine Vorverbindung und fehlt in der OPEN-Anweisung der FILE-Parameter, wird eine aktive Dateiverbindung mit einer temporären Datei erzeugt (siehe Abschnitt 6.2.).
Einheiten und Dateien, die noch nicht oder nicht mehr Bestandteile von aktiven Dateiverbindungen sind, können jederzeit neue aktive Verbindungen eingehen.

6.1. Dateistruktur und Dateizugriff

Dateien haben im Betriebssystem SCP keine durch irgendein Aufzeichnungsverfahren bedingte Satzstruktur. Eine Datei ist also nur eine Folge von Zeichen (Bytes).
Durch Satztrennzeichen CR LF oder durch die Verwendung des RECL-Parameters in einer OPEN-Anweisung kann einer Datei eine Satzstruktur aufgeprägt werden. Durch den RECL-Parameter wird die Satzlänge nur für die Dauer der aktiven Dateiverbindung festgelegt.
Die Aufprägung einer Satzstruktur durch Satztrennzeichen ist nur für Dateien mit formatisierten Sätzen möglich. Die Satzstruktur, die durch Verwendung des RECL-Parameters aufgeprägt wird, hat jedoch Vorrang vor der durch Satztrennzeichen aufgeprägten Satzstruktur. Die Satzlänge für formatisierte Sätze kann auch für sequentiellen Zugriff festgelegt werden. Als Satztrennzeichen werden folgende Zeichen oder Zeichenfolgen von einem FORTRAN77-Programm erzeugt oder erkannt:
CR LF, FF, FF CR, CR FF, CR und CR CR.
Diese Zeichen dürfen deshalb in den formatisierten Sätzen selbst nicht auftreten. Sind sie in auszugebenden Zeichenketten enthalten, wird die Satzstruktur verfälscht.
Die Struktur einer Datei, die von einem FORTRAN77-Programm mittels PRINT- oder WRITE-Anweisungen (mit Angabe des FMT-Parameters) erzeugt wurde, ist von der Eigenschaft PRINT der Datei abhängig.
Eine Datei hat die Eigenschaft PRINT, wenn ihre Sätze auf das Terminal oder den Drucker ausgegeben werden sollen (gültige Dateispe-

zifikationen CON:, X:, LST: und Y:) oder wenn die Datei in ihrem Namen einen der folgenden Dateitypen hat:
LS?, MAP, MP?, PRT, PUN.
Statt des Zeichens ? kann ein beliebiges Zeichen (jedoch kein Leerzeichen) stehen.
Bei der Ausgabe von formatisierten Sätzen in eine Datei mit der Eigenschaft PRINT wird das erste Zeichen jedes Satzes in ein Satztrennzeichen umgewandelt. Damit wird die Bedeutung des ersten Zeichens als Steuerzeichen für einen Drucker realisiert. Eine Datei mit der Eigenschaft PRINT beginnt also immer mit einem Satztrennzeichen.
Bei der Ausgabe von formatisierten Sätzen in eine Datei, die nicht die Eigenschaft PRINT hat, wird das erste Zeichen des Satzes nicht verändert. Als Satztrennzeichen wird die Zeichenfolge CR LF zu jedem Satz hinzugefügt.
Bei der Eingabe einer Datei mit der Eigenschaft PRINT wird das erste Zeichen jedes gelesenen Satzes aus dem Satztrennzeichen erzeugt. Hat die Datei nicht die Eigenschaft PRINT, werden alle Zeichen eingelesen, die zwischen den Satztrennzeichen stehen oder vor der Betätigung der CR-Taste des Terminals eingegeben werden.
In unformatisierten Sätzen sind alle Zeichen möglich. Deshalb können keine Zeichen als Satztrennzeichen verwendet werden. Eine Satzstruktur ist nur durch Festlegung einer Satzlänge zu erreichen. Darum ist die Verwendung des RECL-Parameters auch für sequentiellen Zugriff erlaubt.
Besteht eine Datei aus unformatisierten Sätzen und wurde die Dateiverbindung ohne den RECL-Parameter erzeugt, so hat die Datei keinerlei Satzstruktur. Das bedeutet, daß für solch eine Datei keine BACKSPACE-Anweisung ausgeführt werden kann. Das bedeutet außerdem, daß die durch eine E/A-Anweisung einzulesende oder auszugebende Datenmenge nur durch Anzahl und Typ der Datenlistenelemente bestimmt wird.
Auf Dateien mit formatisierten als auch auf Dateien mit unformatisierten Sätzen kann der Zugriff sequentiell und direkt erfolgen.
Tabelle 6 enthält eine Übersicht, welche Parameterangaben zu welcher Zugriffsart führen.

Tabelle 6 Dateizugriffsarten

RECL-Parameter	ACCESS-Parameter	Zugriffsart
angegeben	nicht angegeben DIRECT	direkt
angegeben	SEQUENTIAL	sequentiell
nicht angegeben	nicht angegeben	

Die Angabe der Satzlänge durch den RECL-Parameter ist also für den Direktzugriff notwendig und für den sequentiellen Zugriff zur Aufprägung einer Satzstruktur möglich. Der Direktzugriff zu einer Datei erfolgt über die relative Adresse eines Satzes in der Datei. Sie berechnet sich aus Satznummer (REC-Parameter) und Satzlänge (RECL-Parameter) wie folgt:

Relative Adresse des Satzes = (Satznummer-1) * (Satzlänge+n)

Durch den Wert n wird bei der Berechnung der relativen Adresse des Satzes die reale Länge der Sätze berücksichtigt.
Dabei gilt:

n = 0 - unformatisierte Datei
n = 1 - PRINT-Datei
n = 2 - nicht PRINT-Datei

Der direkte Zugriff auf eine Datei ist nur möglich, wenn sich die Datei auf einer Diskette befindet.
Beim Direktzugriff auf eine Datei mit formatisierten Sätzen mit einer READ- oder WRITE-Anweisung kann durch "/"-Formatelemente die Verarbeitung mehrerer Sätze gefordert werden.

6.2. Implementierungsbedingte Besonderheiten der FORTRAN77-E/A

Abschließende Leerzeichen eines Satzes einer formatisierten Datei werden nicht ausgegeben, wenn für die Datei kein RECL-Parameter angegeben wurde und wenn die Ausgabe in eine Datei erfolgt, die die Eigenschaft PRINT hat.
Wird ein derart verkürzter Satz mit einer READ-Anweisung und einem oder mehreren A-Formatelementen gelesen, wird er immer als mit Leerzeichen aufgefüllt betrachtet.

Bei der Ausführung eines FORTRAN77-Programms können maximal 6 Dateiverbindungen gleichzeitig aktiv sein.
Eine Datei auf einer Diskette kann nicht mehrere aktive Dateiverbindungen gleichzeitig haben, das heißt, der parallele Zugriff über mehrere UNIT-Nummern auf eine Diskettendatei ist nicht möglich. Dagegen können mit den Geräten Terminal (CON:, X:) und Drucker (LST:, Y:) mehrere FORTRAN77-Dateien über ihre UNIT-Nummer gleichzeitig aktiv verbunden sein.

Als UNIT-Nummern im UNIT-Parameter sind nur die Nummern 0 bis 16 implementiert.
Der UNIT-Parameter der Form UNIT=* ist in einer READ-Anweisung gleichbedeutend mit UNIT=5, in einer WRITE-Anweisung mit UNIT=6.

Temporäre Dateien erhalten einen Dateinamen der Form:

WORKn.TMP

Dabei kann n die Werte 1 bis 6 annehmen. Eine temporäre Datei existiert physisch zwischen der Ausführung der zugehörigen OPEN- und CLOSE-Anweisung.

Bei Ausführung einer READ-Anweisung für eine UNIT, die mit dem Terminal (CON: oder X:) verbunden ist, wird für die Aufforderung zur Eingabe ein Doppelpunkt (gefolgt von einem Leerzeichen) auf die nächste Zeile ausgegeben. Weitere notwendige Eingaben werden bei Verwendung der listgesteuerten Formatisierung durch ein Pluszeichen angefordert.
Endete jedoch die vorhergehende Ausgabe auf das Terminal mit einem Doppelpunkt, wird zur Eingabeanforderung nur ein Leerzeichen an den ausgegebenen Satz angefügt. Weitere Eingaben können ebenfalls auf der gleichen Zeile erfolgen, solange das Steuerzeichen aus einer WRITE- oder PRINT-Anweisung keinen Vorschub auf eine neue Zeile bewirkt.

Zur Eingabe von der Tastatur verwendet ein FORTRAN-Programm die gleiche Betriebssystemfunktion, die zur Eingabe von Kommandos und zum Aufruf von Programmen verwendet wird. Dadurch hat zum Beispiel die Angabe von CTRL-P, CTRL-S und CTRL-Q die gleiche Wirkung wie bei der Eingabe von Kommandos. Die Eingabe von CTRL-C bewirkt den sofortigen Abbruch des Programms.
Das Dateiende kann bei der Eingabe über die Tastatur
- durch Betätigen der "EOF"-Taste,
- durch Eingabe von CTRL-Z sowie
- durch eine Leereingabe (ENTER-Taste ohne vorheriges Betätigen einer anderen Taste)

dem FORTRAN77-Programm angezeigt werden. Im letzteren Fall muß die Gültigkeit der Leereingabe nochmals bestätigt werden.

Wird eine WRITE-Anweisung für eine sequentielle Diskettendatei ausgeführt und die aktuelle Dateiposition ist nicht das Dateiende, so erfolgt die Verkürzung der Datei durch folgende Schritte:
- Umspeichern der verbleibenden Sätze in eine Hilfsdatei,
- Löschen der zu verkürzenden Datei,
- Erzeugen einer Datei mit dem alten Namen,
- Umspeichern der Hilfsdatei in diese Datei,
- Löschen der Hilfsdatei.

Die Hilfsdatei wird auf der Diskette im Standardlaufwerk angelegt. Auf dieser Diskette muß der geeignete Platz vorhanden sein und der Schreibzugriff zu dieser Diskette muß gesichert sein.

Bei der Ausführung einer E/A-Anweisung mit IOSTAT-Parameter erhält die im IOSTAT-Parameter angegebene Variable die in Tabelle 7 zusammengestellten Werte.

Tabelle 7 IOSTAT-Werte

IOSTAT-Wert	Bedeutung
-1	EOF ist aufgetreten
0	Anweisung wurde fehlerfrei ausgeführt
Wert positiv	Fehlercode (siehe Anlage 3)

7. Gleitkomma-Arithmetik in FORTRAN77

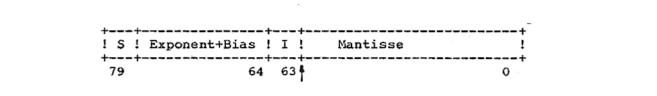
Der FORTRAN77-Compiler für SCP benutzt für die Realisierung der Gleitkomma-Arithmetik den in [6] beschriebenen Emulator.
In diesem Abschnitt werden nur die Informationen zu diesem Emulator geliefert, die für das Verständnis der FORTRAN77-spezifischen Anwendung notwendig sind. Weitere Details sind [6] zu entnehmen.
Die Semantik der durch den FORTRAN77-Compiler erzeugten Gleitkomma-Befehle ist in [3] beschrieben.

7.1. Gleitkomma-Emulator

Für die internen Operationen verwendet der Emulator einen speziellen Datentyp (im folgenden TEMPREAL genannt) mit folgendem Gültigkeitsbereich:

$$3.4*10^{**-4932} < |x| < 1.2*10^{**4932} \quad \text{oder} \quad |x| = 0$$

Ein TEMPREAL hat folgenden internen Aufbau:



S : Vorzeichenbit
↑ : Position des impliziten Kommas
I : erstes Bit der Mantisse
Bias : 16383 (3FFFH)

Die Befehle des Emulators gestatten es, auch explizit mit Operanden vom Datentyp TEMPREAL zu arbeiten. Bei der Realisierung der Standardfunktionen von FORTRAN77 wird dieser Datentyp verwendet, um möglichst wenig Genauigkeitsverluste zu erhalten. Zu diesem Zweck werden die Argumente vor der Verarbeitung in TEMPREAL-Argumente konvertiert. Ebenso wird das TEMPREAL-Ergebnis in den Zieldatentyp umgewandelt.

Die wesentlichen Informationen zur Steuerung des Emulators und seinen gegenwärtigen Zustand werden in einem Steuerwort (Control Word=CW) bzw. einem Statuswort (SW) gehalten.

Das CW wird benutzt, um den Emulator an bestimmte Anwendungsbedingungen anzupassen. Es hat folgenden Aufbau:

Masken für Ausnahmebedingungen:		Bit
INVALID OPERATION (1)		0
DENORMALIZED OPERAND (1)		1
ZERODIVIDE (1)	(5)	2
OVERFLOW (1)		3
UNDERFLOW (1)		4
PRECISION (1)		5
Interruptmaske (1)		7
Precision-Modus (2)		8-9
Rundungs-Modus (3)		10-11
Infinity-Modus (4)		12

Die Bits 6 und 13-15 sind nicht signifikant.

(1) Bedeutung der Bitbelegung:
0 = nicht maskiert oder Interrupt aktiv
1 = maskiert oder Interrupt nicht aktiv

(2) Bedeutung der Bitbelegung:
00 = 24 Bits Mantisse
01 = reserviert
10 = 53 Bits Mantisse
11 = 64 Bits Mantisse

(3) Sei Z das mathematisch exakte Ergebnis einer arithmetischen Operation und Z1,Z2 die im Zieldatentyp exakt darstellbaren Nachbarn von Z mit $Z1 < = Z < = Z2$, so gilt:

00 = die Zahl von Z1 und Z2, die Z am nächsten liegt; falls !Z1-Z! = !Z2-Z!, so wird von Z1 und Z2 die Zahl gewählt, deren letztes Bit 0 ist.
01 = Z1 (Abrunden gegen $-\infty$)
10 = Z2 (Aufrunden gegen $+\infty$)
11 = die Zahl von Z1 und Z2, die näher zu Null liegt (Abschneiden)

(4) Bedeutung der Bitbelegung:
0 = projektiv ($+\infty$ und $-\infty$ werden gleich behandelt)
1 = affin ($+\infty$ und $-\infty$ werden unterschieden)

(5) Wenn eines dieser Bits auf 0 gesetzt ist, muß auch Bit 7 (Interruptmaske) auf 0 gesetzt sein, damit beim Auftreten einer nicht maskierten Ausnahmebedingung eine Interruptbehandlung angefordert wird.

Das Statuswort (SW) enthält Informationen über den aktuellen Zustand des Emulators. Es hat folgenden Aufbau:

	Bit
INVALID OPERATION (1)	0
DENORMALIZED OPERAND (1)	1
ZERODIVIDE (1)	2
OVERFLOW (1)	3
UNDERFLOW (1)	4
PRECISION (1)	5
Unterbrechungsanforderung (2)	7
Rückkehrcode (3)	8-10, 14
Stackpointer (4)	11-13
(5)	15

(1) Ist das entsprechende Bit mit 1 belegt, so ist die Ausnahmebedingung aufgetreten.

(2) Ist dieses Bit mit 1 belegt, so ist eine Unterbrechung aufgetreten (bei FORTRAN77 nur im Falle "INVALID OPERATION" möglich)

(3) Der Rückkehrcode wird für bedingtes Verzweigen benötigt.

(4) Gibt das aktuelle erste Element des Gleitkomma-Stacks des Emulators an.

(5) Eine Belegung mit 1 gibt die Abarbeitung eines Gleitkomma-Befehls an.

Das Statuswort wird in die FORTRAN77-Laufzeitfehlermeldungen einbezogen. Für die FORTRAN77-Anwender sind nur die Bits 0-5 wesentlich, die die aufgetretenen Ausnahmebedingungen anzeigen.

Während der Laufzeitinitialisierung von FORTRAN77 wird das CW wie folgt belegt:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0

Damit werden folgende Festlegungen getroffen:

- Der Infinity-Modus ist projektiv, d.h. $+\infty$ und $-\infty$ werden nicht unterschieden ($+\infty$ und $-\infty$ werden im Emulator durch Spezialwerte dargestellt, die als Standardergebnis bei einer OVERFLOW-Ausnahmebedingung entstehen können).
- Für alle impliziten Rundungen (außer bei der Konvertierung von REAL zu INTEGER) wird der Rundungsmodus mit der Bitbelegung 00 (siehe Bild 2) gewählt. Eine Rundung wird durchgeführt, wenn ein Resultat nicht im gewünschten Zieldatentyp exakt darstellbar ist.
- Alle Gleitkomma-Zwischenresultate werden mit einer Genauigkeit von 64 Bits berechnet.

- Bei allen Ausnahmebedingungen, außer "INVALID OPERATION", werden die Standardmaßnahmen des Emulators wirksam. Bei "INVALID OPERATION" erfolgt ein Abbruch der Abarbeitung des Programms mit einer Fehlermeldung.
  Die Standardmaßnahmen werden im folgenden Abschnitt erläutert.

- Die Unterbrechungsmaske steht auf 0, so daß beim Auftreten von "INVALID OPERATION" stets die im vorangehenden Anstrich beschriebene Aktion wirksam wird.

7.2. Ausnahmebedingungen des Gleitkomma-Emulators

Im folgenden werden die Maßnahmen beschrieben, die beim Auftreten einer Ausnahmebedingung während der Ausführung eines FORTRAN77-Programms gewählt werden, und es werden die Bedingungen angegeben, unter denen diese Ausnahmebedingungen auftreten können. Für die Beschreibung der Maßnahmen sind folgende Begriffserläuterungen notwendig:

normalisierter Wert:
Eine normale Null oder ein Wert, dessen führendes Mantissen-Bit 1 und dessen Exponent größer als Null ist.

denormalisierter Wert:
Ein Wert, dessen Exponent gleich 0 ist und dessen führendes explizites oder implizites Mantissen-Bit 0 ist, aber keine normale Null darstellt.

unnormalisierter Wert:
Ein solcher Wert tritt nur im TEMPREAL-Format auf und entsteht aus einem denormalisierten Wert. Er ist aufgebaut wie ein normaler TEMPREAL-Wert, nur daß das führende Bit der Mantisse 0 ist. Mit unnormalisierten Werten kann eine normale TEMPREAL-Arithmetik durchgeführt werden. Es wird aber stets durch die 0 im führenden Mantissen-Bit angezeigt, ob bei der Berechnung Genauigkeitsverluste aufgetreten sind, die sich auf die darstellbaren Mantissen-Stellen auswirken.

INVALID OPERATION

Beim Auftreten dieser Ausnahmebedingung kommt es zu einer Unterbrechung, die nach Ausgabe einer Fehlermeldung zum Abbruch des Programms führt.
Eine solche Ausnahmebedingung tritt auf, wenn entweder ein Operand für eine bestimmte Operation nicht erlaubt ist oder die Operation selbst ungültig ist.
Diese Ausnahmebedingung weist im allgemeinen auf einen Programmierfehler hin, wie z.B. die Bezugnahme auf eine nicht initialisierte Variable.

Unter folgenden Bedingungen kann diese Ausnahmebedingung bei der Abarbeitung eines FORTRAN77-Programms auftreten:

- Einer oder mehrere Operanden einer Folge von Operationen sind unnormalisiert oder denormalisiert und ein gültiges Resultat kann nicht garantiert werden, da nicht vertretbare Genauigkeitsverluste entstehen würden.

- eine der folgenden Operationen wird versucht:

```
                    unendlich + unendlich
                    unendlich - unendlich
                      0.0     * unendlich
                    unendlich *      0.0
                    unendlich / unendlich
                          0.0 / 0.0
          normalisierte Zahl / unnormalisierte Zahl
          normalisierte Zahl / denormalisierte Zahl
```

- einer der folgenden Vergleiche wird versucht:

```
                  unendlich mit     0.0
                  unendlich mit normalisierter Zahl
                  unendlich mit unendlich
```

- Bei INT oder NINT ist das Argument zu groß, um in das INTEGER-Format zu passen

- SQRT(x), wenn x eine negative Zahl, eine denormalisierte Zahl, eine unnormalisierte Zahl oder +- unendlich ist

- SIN(x), COS(x), TAN(x), EXP(x), wenn x = +- unendlich oder !x! eine unnormalisierte Zahl >= 2 ** -63 ist

- ASIN(x), ACOS(x), wenn x = +- unendlich oder !x! eine unnormalisierte Zahl >= 2 ** -63 oder !x! >= 1 ist

- ATAN(x), wenn !x! eine unnormalisierte Zahl >= 2 ** -63 ist

- LOG(x), LOG10(x), wenn 0 > x oder x eine unnormalisierte oder denormalisierte Zahl oder +- unendlich ist

- SINH(x), COSH(x), TANH(x), wenn !x! eine unnormalisierte Zahl >= 2 ** -63 ist

- ATAN2(y,x), wenn x und y unnormalisierte Zahlen sind und !y/x! >= 2 ** - 63 gilt oder wenn !x! = !y! = 0 oder !x! = !y! = unendlich

- MIN(x1,x2,...,xn), MAX(x1,x2,...,xn), wenn eines der Argumente unendlich ist

- AMOD(x,y), DMOD(x,y), wenn y = +- unendlich oder x ist unnormalisiert oder denormalisiert

- DIM(x,y), wenn x = y = +- unendlich

- y ** x, wenn eine der folgenden Bedingungen erfüllt ist:
  y = x = +- 0
  y normalisiert, x = +- unendlich
  y = - unendlich, x unnormalisierte Zahl oder +- unendlich oder +- 0
  y = + unendlich, x = +- 0
  y = + unendlich, x = + unendlich
  y = +- 0 , x = +- unendlich
  y = + unendlich, x = - unendlich
  y = + unendlich, x normalisiert (nicht ganzzahlig)
  y = - unendlich, x normalisiert (nicht ganzzahlig)
  y < 0 normalisiert, x normalisiert (nicht ganzzahlig)

- y ** i, wenn i < 0, i sich nicht in eine ganze Zahl von 32 Bit Länge umwandeln läßt und y eine unnormalisierte Zahl ist

DENORMALIZED OPERAND

Beim Auftreten dieser Ausnahmebedingung kommt es zu keiner Programmunterbrechung, da sie für FORTRAN77 maskiert ist.
Diese Ausnahmebedingung tritt auf, wenn einer oder mehrere der Operanden denormalisierte Zahlen sind. Eine denormalisierte Zahl wird standardmäßig als Reaktion auf eine "Underflow"-Ausnahmebedingung geliefert.
Die denormalisierten Operanden werden in entsprechende unnormalisierte Operanden umgewandelt, und es wird anschließend die korrekte Arithmetik für unnormalisierte Operanden ausgeführt.
Tritt ein denormalisierter Operand als Argument von Standardfunktionen auf, so werden die in der folgenden Liste angegebenen Resultate geliefert oder Aktionen ausgeführt:

Funktion	Resultat
ACOS(x)	$\pi/2$
ASIN(x)	x
SINH(x)	x
COSH(x)	1
SIN(x)	x
COS(x)	1
EXP(x)	1
ATAN(x)	x
LOG(x)	INVALID OPERATION
LOG10(x)	INVALID OPERATION
SQRT(x)	INVALID OPERATION
TAN(x)	x
TANH(x)	x
ATAN2(y,x)	Denormalisierte Argumente werden in normalisierte umgewandelt und die Funktionsberechnung wird fortgesetzt.
DIM(y,x)	Reaktion wie bei ATAN2
MAX(x1,x2,...,xn)	Reaktion wie bei ATAN2
MIN(x1,x2,...,xn)	Reaktion wie bei ATAN2
SIGN(y,x) x denormalisiert	Vorzeichen wird von x übernommen
MOD(y,x) x denormalisiert	INVALID OPERATION

ZERODIVIDE

Die Ausnahmebedingung "ZERODIVIDE" tritt auf, wenn der Divisor einer Divisionsoperation gleich Null und der Dividend eine endliche Zahl ungleich Null ist.
Als Resultat wird "unendlich" geliefert, wobei sich das Vorzeichen aus den Operandenvorzeichen ergibt. Eine Programmunterbrechung erfolgt nicht.
"ZERODIVIDE" wird durch die FORTRAN77-Bibliothek auch gesetzt, wenn folgende Fälle auftreten:

- y ** x, y ** i, wenn y=0 und x < 0 bzw. i < 0 (x, y reell, i ganzzahlig)
- TAN(x), wenn x = ungeradzahliges Vielfaches von $\pi/2$
- LOG(x), LOG10(x), wenn x=0

OVERFLOW

Die Ausnahmebedingung "OVERFLOW" tritt auf, wenn ein gerundetes Resultat endlich ist, aber sein Exponent für den Resultatdatentyp zu groß ist.
Diese Ausnahmebedingung führt zu keiner Programmunterbrechung. Das Resultat wird gleich "unendlich" gesetzt und die PRECISION-Ausnahmebedingung im Steuerwort angezeigt.
Für EXP(x), SINH(x), COSH(x), y ** x und y ** i wird die "Overflow"-Ausnahmebedingung durch die FORTRAN77-Bibliothek nach Prüfung der Argumente bereits vor Ausführung der Funktionsberechnung gesetzt.

UNDERFLOW

Die Ausnahmebedingung "UNDERFLOW" tritt auf, wenn eine der folgenden Bedingungen erfüllt ist:

- Ein gerundetes Resultat einer Operation hat einen für den Resultattyp zu kleinen Exponenten

- Das Ergebnis einer Multiplikation oder Division, deren Operanden verschieden von Null sind, ist nicht von Null unterscheidbar

Das Resultat nach Auftreten der Ausnahmebedingung "UNDERFLOW" ist eine korrekt gerundete denormalisierte Zahl oder Null.
Eine Programmunterbrechung tritt nicht auf.

PRECISION

Die Ausnahmebedingung "PRECISION" tritt auf, wenn das korrekt gerundete Resultat einer Operation vom ungerundeten Resultat verschieden ist oder in dieser Operation vorher eine "OVERFLOW"-Ausnahmebedingung aufgetreten ist.
Als Ergebnis wird stets das korrekt gerundete Resultat geliefert.
Eine Programmunterbrechung tritt nicht auf.

8. Fehlermeldungen

8.1. Fehlermeldungen zur Übersetzungszeit

Zur Übersetzungszeit werden die lexikalischen, syntaktischen und semantischen Fehler innerhalb einer FORTRAN77-Programmeinheit durch die Pässe des Compilers erkannt und in der Fehlerdatei name.ERR aufgelistet. Eine vollständige Liste aller Übersetzungsfehlermeldungen ist in Anlage 2 enthalten.
Ist die Option Q (siehe Abschnitt 3.4.1.) unwirksam, erfolgt zusätzlich und unmittelbar bei Erkennung des Fehlers die Ausgabe des Fehlers auf das Terminal in folgender Kurzform:

```
Error:   nr, Sev: schw, Stmt: anr, LINE: znr, Insert: einfügung
```

Dabei entsprechen die Angaben nr, schw, anr und znr genau den zugehörigen Spalten in der Fehlerdatei. Unter Insert ist die Einfügung angegeben, die im Fehlertext der Anlage 2 mit § spezifiziert ist. Nach der Kurzform einer Fehlermeldung erscheint die Ausschrift:

```
Enter E or CTRL-C to abort or any other to continue
```

E als Antwort bewirkt ein sofortiges Verzweigen in den EPASS des Compilers zur Ausgabe aller bisher erzeugten Fehlermeldungen und anschließende Beendigung der Übersetzung. Bei CTRL-C wird die Übersetzung sofort abgebrochen.
Nach erfolgter Übersetzung kann zum Beispiel mit dem Kommando TYPE die Fehlerdatei ausgegeben werden. Sie hat folgenden Inhalt:

```
Error diagnostics

Error     Sev       Stmt      LINE      Message

'nr'      'schw'    'anr'     'znr'     'meldungstext mit einfügung'
 .         .         .         .          .
 .         .         .         .          .
 .         .         .         .          .
```

Dabei ist nr die Fehlernummer, anr die Nummer und znr die Anfangszeilennummer der fehlerhaften Anweisung.
Kann einem Fehler keine Anweisung zugeordnet werden, wird als Anweisungsnummer Null ausgegeben.

In der Spalte Sev enthält die Fehlerdatei wie auch die Kurzform einer Fehlermeldung die Fehlerschwere des aufgetretenen Fehlers. Sie hat folgende Bedeutung:

Fehlerschwere	Bedeutung
T	Abbruchfehler; Der Fehler ist so schwerwiegend, daß eine Fortsetzung der Übersetzung unmöglich ist.
S	Schwerer Fehler; Der Fehler ist so schwer, daß eine Codeerzeugung (CPASS) nicht sinnvoll ist. Die Übersetzung bricht nach der globalen Stufe (GXPASS) ab. Die Codeerzeugung kann aber mit der Option COMPILE erzwungen werden. Dabei wird die Anweisung gestrichen und durch einen Aufruf des Laufzeitfehlerbehandlers ersetzt.
E	Fehler; Der aufgetretene Fehler kann noch zu einer sinnvollen Codeerzeugung führen. Falls das nicht gewünscht wird, kann durch NOCOMPILE(E) auch in diesem Fall der Abbruch nach der globalen Stufe erreicht werden.
W	Warnung; Der Fehler ist von geringer Bedeutung. Der Übersetzungsprozeß wird nicht beeinträchtigt.
I	Information.

Während der Übersetzungszeit kann auch eine weitere Art von Fehlermeldungen entstehen, die auf Fehler im Compiler selbst zurückzuführen sind. Sie erscheinen auf dem Terminal und haben die Form:

```
Compiler error: 'nr'   stmt: 'anr'   line: 'lnr'   id: 'kennz'
```

oder

```
Compiler error: 'nr'
```

Bei Auftreten dieser Fehlerart ist über die zuständige Betreuungsgruppe Kontakt mit dem Entwickler aufzunehmen. Kommt es während einer Übersetzung zu Fehlerzuständen bei der Speicherplatzinitialisierung oder bei der Arbeit mit den Compilerdateien, so erscheint auf dem Terminal eine selbsterklärende Meldung.

8.2. Fehlermeldungen zur Ausführungszeit

Alle Fehlermeldungen zur Ausführungszeit werden auf dem Terminal ausgegeben. Dabei werden folgende Fehlergruppen unterschieden:

- E/A-Fehler
- FORTRAN77-E/A-Fehler
- Festkomma- und Überlauffehler
- Fehler bei der Abarbeitung von Gleitkomma-Funktionen
- Stop durch Übersetzungsfehler
- Allgemeine Fehler

In den folgenden Abschnitten werden die Meldungen dieser Fehlerarten beschrieben. Dabei bedeuten:

'hexnr' - Hexadezimale Fehlernummer. In Anlage 3 ist eine vollständige Liste aller Ausführungszeit-Fehlernummern und ihre Bedeutung zusammengefaßt.

'hexbasis':
'hexoffs' - Hexadezimale Adresse, bei der der Fehler aufgetreten ist. Dabei entsteht die Adresse aus 'hexbasis'*16+'hexoffs'.

'pname' - Name der Programmeinheit, in der der Fehler auftrat.

'deznr' - Dezimale Nummer einer Anweisung oder eines Übersetzungsfehlers.

'statwort' - Statuswort des Gleitkomma-Emulators (siehe Abschnitt 7.).

'fktname' - Name einer FORTRAN77-Standardfunktion oder einer mathematischen Operation (siehe Tabelle 8)

'cd' - Hexadezimaler Operationscode eines Gleitkomma-Befehls

'addr' - Hexadezimale Adresse

Tabelle 8 Einfügungen in Fehlermeldungen zu Gleitkomma-Funktionen

Einfügung	Bedeutung
MAX	FORTRAN77-Standardfunktion
MIN	FORTRAN77-Standardfunktion
ROUND	FORTRAN77-Standardfunktion
TRUNC	FORTRAN77-Standardfunktion
Y**I	Potenzierung; Exponent INTEGER
INT	FORTRAN77-Standardfunktion
ARCTAN(Y,X)	FORTRAN77-Standardfunktion
ARCTAN	FORTRAN77-Standardfunktion
ARCCOS	FORTRAN77-Standardfunktion
ARCSIN	FORTRAN77-Standardfunktion
TAN	FORTRAN77-Standardfunktion
COS	FORTRAN77-Standardfunktion
SIN	FORTRAN77-Standardfunktion
TANH	FORTRAN77-Standardfunktion
COSH	FORTRAN77-Standardfunktion
SINH	FORTRAN77-Standardfunktion
LOG10	FORTRAN77-Standardfunktion
LN	FORTRAN77-Standardfunktion
EXP	FORTRAN77-Standardfunktion
Y**X	Potenzierung; Exponent REAL
MOD	FORTRAN77-Standardfunktion
MINT	FORTRAN77-Standardfunktion
ANINT	FORTRAN77-Standardfunktion
AINT	FORTRAN77-Standardfunktion
DIM	FORTRAN77-Standardfunktion
SIGN	FORTRAN77-Standardfunktion
CSQRT	FORTRAN77-Standardfunktion
CEXP	FORTRAN77-Standardfunktion
CLOG	FORTRAN77-Standardfunktion
CSIN	FORTRAN77-Standardfunktion
CCOS	FORTRAN77-Standardfunktion
C**C	Potenzierung; Exponent Complex

8.2.1. E/A-Fehlermeldungen

Auf dem Terminal erscheint folgende Meldung:

```
*** Run-time I/O exception 'hexnr'
*** Near location 'hexbasis':'hexoffs'
*** In program 'pname' statement 'deznr'   ])
*** Job aborted.
```

8.2.2. FORTRAN77-E/A-Fehlermeldungen

Die Meldung auf dem Terminal hat folgendes Aussehen:

```
*** Run-time FORTRAN77 I/O exception 'hexnr'
*** Near location 'hexbasis':'hexoffs'
*** In program 'pname' statement 'deznr'   ])
*** Job aborted.
```

8.2.3. Festkomma- und Überlauffehler

Folgende Meldung erscheint auf dem Terminal:

```
*** Run-time 'text' exception
*** Near location 'hexbasis':'hexoffs'
*** In program 'pname' statement 'deznr'   1)
*** Job aborted.
```

Dabei steht für 'text' eine der folgenden Zeichenketten:

INTEGER-OVERFLOW	Festkomma-Überlauf
INTEGER-ZERO-DIVIDE	Festkomma-Nulldivision
RANGE	Überlauf von Feldgrenzen oder Teilkettengrenzen

8.2.4. Fehler bei der Abarbeitung von Gleitkomma-Funktionen

Kommt es während der Abarbeitung einer der Routinen der Bibliothek F77ART.L86 (Realisierung von FORTRAN77-Standardfunktionen usw.) zu einem Fehler, so erscheint folgende Fehlermeldung auf dem Terminal:

```
*** Run-time 'fktname' exception 'statwort'
*** Near location 'addr'
*** In program 'pname' statement 'deznr'   1)
*** Job aborted.
```

8.2.5. Gleitkomma-Fehler

Fehler bei der Abarbeitung von Gleitkomma-Befehlen außerhalb der Routinen der Bibliothek F77ART.L86 führen zu folgender Form der Fehlermeldung auf dem Terminal:

```
*** Run-time floating-point exception 'statwort'
*** Instr opcode 'cd'
*** Memop address 'addr'
*** Near location 'addr'
*** In program 'pname' statement 'deznr'   1)
*** Job aborted.
```

8.2.6. Allgemeine Fehler

Alle anderen Fehler, die nicht näher klassifiziert werden können, erscheinen in folgender Form auf dem Terminal:

```
*** Run-time exception 'hexnr'
*** Near location 'hexbasis':'hexoffs'
*** In program 'pname' statement 'deznr'   1)
*** Job aborted.
```

8.2.7. Stop durch Übersetzungszeitfehler

Dieser Abbruch der Ausführung wird wie folgt auf dem Terminal protokolliert:

```
*** Run-time exception
*** Compilation error 'deznr'
*** In program 'pname' statement 'deznr'   1)
*** Job aborted.
```

1) ** in program 'pname', wenn Option NOGOSTATEMENT für die Übersetzung angegeben ist.

9. Optimierung von FORTRAN77-Programmen

Die logische Stufe Optimierung erzeugt effektiven Zwischencode, wenn eine entsprechende Option wirksam ist und kein vorzeitiger Abbruch der Übersetzung durch einen Fehler oder eine Abbruchbedingung erfolgte. Voraussetzung für eine Optimierung ist eine Kennzeichnung von Programmabschnitten einer Programmeinheit durch eine Vorbereitungsstufe, die über die Art des Betretens und Verlassens der Programmabschnitte Auskunft gibt.

9.1. Optimierungsstufe 1 (OPT=1)

Ein wesentlicher Teil der Optimierung ist die Ausdrucksoptimierung.
Die Ausdrucksoptimierung erfaßt arithmetische und logische Ausdrücke, auch solche, die in Feldelement-Adreßberechnungen auftreten, sowie die Argumentlisten von Unterprogrammaufrufen.

Treten im Programm formal gleiche Ausdrücke auf und wird der Wert der in den Ausdrücken enthaltenen Variablen zwischen den Berechnungen der Ausdrücke nicht geändert, so wird nur der im Ablauf erste Ausdruck berechnet und dessen Wert anstelle der folgenden gleichen Ausdrücke verwendet.
Ausdrücke, deren Operanden nur Konstanten sind, werden zur Übersetzungszeit berechnet.
Wird eine Variable im Programm nur mit einem Wert belegt und nicht verwendet, so wird die zugehörige Anweisung gestrichen.
Wird eine Variable im Programm nur einmal mit einem Wert belegt, wird sie bei der Verwendung in Ausdrücken wie eine Konstante behandelt.
Bei der Feldelement-Adreßberechnung tritt nicht nur für gleiche Feldelemente, sondern auch für Elemente gleich aufgebauter Felder und für unterschiedlich strukturierte Felder mit gleichen Indexausdrücken ein Optimierungseffekt auf.
Argumentlisten werden schon dann optimiert, wenn sie in mindestens einem Argument übereinstimmen.

Arithmetische Operatoren werden auf möglichst niedriges Niveau zurückgeführt, um eine Verkürzung der Ausführungszeit zu erreichen (z.B. A ** 2 --> A * A).
Operationen, die unabhängig vom Wert der Variablen nur zu einem Ergebnis führen, werden von der Optimierungsstufe ausgeführt (z.B. A ** 0 --> 1).

Außer der oben beschriebenen Ausdrucksoptimierung wird innerhalb von DO-Schleifen eine Codeverbesserung durch Verlagerung schleifeninvarianter Teilausdrücke innerhalb arithmetischer Ergibtanweisungen in den Schleifenkopf vorgenommen. Dadurch werden solche Ausdrücke einer unnötigen wiederholten Berechnung entzogen. Dies erstreckt sich auch auf Feldelement-Adreßberechnungen.
Darüberhinaus wird für Ausdrücke, die hinsichtlich der Laufvariablen der DO-Schleife Polynome 1. Grades und ansonsten schleifeninvariant sind, eine inkrementale Funktion gebildet. Dadurch wird bei jedem Schleifendurchlauf nicht der vollständige Ausdruck neu berechnet, sondern sein (gespeicherter) Wert lediglich um den durch die Schrittweite der Laufvariablen hervorgerufenen Änderungsbetrag korrigiert.

9.2. Optimierungsstufe 2 (OPT=2)

Außer dem in Abschnitt 9.1. genannten Optimierungsumfang wird der Code in impliziten Schleifen verbessert. Solche Schleifen sind die durch den Programmierer mittels Sprunganweisungen gebildeten Zyklen. In ihnen werden ebenfalls Verlagerungen von schleifeninvarianten Codeteilen, ähnlich wie in Abschnitt 9.1. beschrieben, vorgenommen.

9.3. Allgemeine Bemerkungen zur effektiven Programmierung

Alle genannten Optimierungen verlängern deutlich die Übersetzungszeit, da dabei mitunter sehr große Programmabschnitte (Schleifen) global zu erfassen sind und eine Vervielfachung des Aufwandes im Falle von mehreren geschachtelten DO- oder impliziten Schleifen entsteht.

Der Programmierer kann bis zu einem gewissen Grade auf die Übersetzungszeiten Einfluß nehmen. Dazu gehört beispielsweise die Vermeidung der Berechnung konstanter Ausdrücke und der mehrfachen Berechnung gleicher Ausdrücke. Darüberhinaus sollte der Programmierer bei Verwendung von Schleifen jeder Art versuchen, schleifeninvariante Teile zu erkennen und durch deren Umordnung vor die Schleifen wiederholter Berechnung zu entziehen. Dadurch wird eine Schleifenoptimierung erleichtert, wenn auch in der Regel nicht überflüssig, da infolge Feldelement-Adreßberechnungen Optimierungsmöglichkeiten entstehen können.
Die für die Optimierungsstufen 1 und 2 erwähnten inkrementalen Funktionen bringen sicher erst Laufzeiteinsparungen, wenn die DO-Schleifen genügend häufig durchlaufen werden.

Anlage 1 Bereichsgrenzen und interne Darstellung der FORTRAN77-Datentypen

Datentyp	Länge(in Bytes)	Bereich
INTEGER*2	2	-32768 ≤ x ≤+32767
INTEGER*4	4	-2147483648 ≤ x ≤+2147483647
REAL*4 1)	4	10**(-37)≤|x|≤10**37
REAL*8 1)	8	10**(-307)≤|x|≤10**307
DOUBLE PRECISION	entspricht REAL*8	
COMPLEX 2)	2*4	Wertbereich für Real- und Imaginärteil entspricht dem REAL*4-Wertebereich
LOGICAL*2	2	FALSE, TRUE
LOGICAL*4	4	FALSE, TRUE
CHARACTER*n	1≤n≤32767	

Erläuterungen:

1) Die interne Darstellung der Real-Datentypen entspricht der gültigen Darstellung der SHORT- bzw. LONG REAL-Datentypen des Gleitkomma-Emulators.

2) Real- und Imaginär-Teil des komplexen Datentyps stehen im Speicher hintereinander und entsprechen der REAL*4-Darstellung.

Die interne Darstellung der LOGICAL-Daten hat folgendes Format:

FFH/00H	undefiniert
1.Byte	1 oder 3 Bytes

00H steht für FALSE
FFH steht für TRUE

CHARACTER-Daten werden intern im ASCII-Code gespeichert.

Anlage 2 Fehlernachrichten zur Übersetzungszeit

22 I Line(s) shorter than 72 characters.
Programmeinheit enthält eine oder mehrere Zeilen, die kürzer als das Standardformat von 72 Zeichen sind.

42 E Error in control line.
Fehler in Steuerkarten.

43 T No FORTRAN text.
Keine FORTRAN-Anweisung gefunden.

90 S Primary and secondary ENTRY incompatible.
Datenattribut und Länge der Eintrittspunkte widersprechen sich.

91 E Line § is not an initial line.
Zeile "§" ist keine Anfangszeile.

92 E Too many digits for label in line §.
Die Marke in Zeile "§" enthält zu viele Ziffern.

93 E Label not complete in line §.
Marke nicht vollständig in Zeile "§".

94 E Line § expects a continuation line.
Nach Zeile "§" wird eine Fortsetzungszeile erwartet.

95 S Statement starts with incorrect character in line §.
Anweisung in Zeile "§" beginnt mit ungültigem Zeichen.

96 E Invalid label in line §.
Ungültige Marke in Zeile "§".

97 E Invalid format in line §.
Ungültiges Format in Zeile "§".

98 E Unknown statement type in line §.
In Zeile "§" beginnt nicht erkennbarer Anweisungstyp.

99 S Analysis of statement impossible in line §.
In Zeile "§" beginnt nicht analysierbare Anweisung.

100 S Unallowed statement starts in line §.
In Zeile "§" beginnt unerlaubte Anweisung.

101 S Statement not permitted in this position.
Anweisung an dieser Stelle unzulässig. Sie wurde ignoriert.

102 S Invalid text after statement end ignored.
Unerlaubter oder nicht interpretierbarer Text nach Anweisungsende wurde ignoriert.

103 E Invalid character. Statement truncated.
Anweisung enthält ungültiges Zeichen. Nachfolgender Text wurde ignoriert.

104 E END statement not in last line.
END-Anweisung ist nicht die letzte Zeile.

105 S First operand LOGICAL.
Erster Operand im Vergleich ist vom Typ LOGICAL.

106 S Ending apostroph assumed at end of statement.
Fehlender Apostroph einer Zeichenkette wurde am Anweisungsende angenommen.

107 S Erroneous relational expression.
Zweiter Operand im Vergleich ist vom Typ LOGICAL oder CHARACTER- und arithmetischer Operand werden verglichen.

108 S ',' does not [immediately] follow an index expression near §.
Fehlerhafter Text im Indexausdruck für Feld "§" wurde ignoriert.

109 W § can not be INTRINSIC because of actual arguments.
Die Datentypen oder die Anzahl der aktuellen Parameter erzwingen, daß "§" der Name einer externen Funktion statt einer INTRINSIC-Funktion ist.

110 S Erroneous type of § operand.
Unzulässiger Typ des "§" Operanden in logischer, arithmetischer oder CHARACTER-Operation.

111 S ')' does not [immediately] follow last index expression near §.
Klammer nach Indexliste für Feld "§" fehlt, oder fehlerhafter Text vor dieser Klammer wurde ignoriert.

112 S Exponent not of type INTEGER.
Exponent im konstanten Ausdruck ist nicht vom Typ INTEGER.

113 S § illegal in context.
Bezeichner "§" ist unpassend zum Kontext.

114 S Illegal concatenation with operand of *-length.
Verkettung außerhalb einer Ergibtanweisung enthält Operanden mit *-Länge.

115 E) inserted after arithmetic expression.
')' wurde nach arithmetischen Ausdruck eingefügt.

116 E Parenthesis or operator illegal.
Klammer oder Operator vor einem Element der Eingabeliste.

117 S Missing or wrong operand/expression.
Fehlender oder falscher Operand im Ausdruck oder fehlender Indexausdruck.

118 E Merging of COMPLEX and DOUBLE PRECISION.
Auftreten eines COMPLEX- und eines DOUBLE PRECISION-Operanden in einer arithmetischen Operation.

119 S Argument list missing.
Argumentliste fehlt in CALL-Anweisung.

120 S Unpaired parenthesis. Statement ignored.
Unpaarige Klammern. Anweisung wurde ignoriert.

129 E Invalid alternate return.
Alternativer Rücksprung ist nicht erlaubt.

130 S Implementation restriction. Too many labels in argument list.
Die Argumentliste enthält zu viele Marken.

131 E Character constant invalid. Blank assumed.
Zeichenkettenkonstante der Länge Null ist nicht erlaubt.
Ein Leerzeichen wurde angenommen.

132 E Character constant truncated.
Zu lange Zeichenkettenkonstante wurde rechtsbündig abgeschnitten.

134 E Missing expression replaced by zero.
Fehlender Ausdruck wurde durch 0 ersetzt.

135 S Implementation restriction. Too many labels in label list.
Zu viele Marken in der Markenliste.

136 S Number § exceeds value limit.
Zahl "§" außerhalb des Wertevorrates.

137 E Statement contains invalid exponent.
Anweisung enthält ungültigen Exponenten.

138 E Number too long.
Zahl hat zu viele Ziffernstellen.

139 E Missing END statement inserted.
Fehlende END-Anweisung wurde eingefügt.

140 S END statement exceeds initial line.
END-Anweisung nicht vollständig auf Anfangszeile.

141 E Statement too long.
Anweisung ist zu lang.

142 E Statement over too many lines.
Anweisung hat zu viele Zeilen.

143 S Label not permitted in this statement.
Anweisung enthält unerlaubte Marke.

144 S Erroneous symbolic name.
Anweisung enthält fehlerhafte Bezeichnung.

145 S Error in name list.
Fehler in der Namenliste.

146 E Error in label list.
Fehler in der Markenliste.

147 E Invalid label list ignored.
Ungültige Markenliste wurde ignoriert.

148 S () assumed.
In der Anweisung wurden () angenommen.

149 E Missing § inserted.
Das fehlende Zeichen "§" wurde eingefügt.

150 S Implementation restriction. Too many elements.
Implementierungsbedingte Einschränkung. FORMAT-Anweisung enthält zu viele Elemente.

151 E Format element § incomplete.
Unvollständiges FORMAT-Element "§".

152 S § empty format element(s).
"§" leere(s) FORMAT-Element(e) wurde(n) ignoriert.

153 E Implementation restriction. Number in § too large.
Implementierungsbedingte Einschränkung. Zahl im FORMAT-Element "§" ist zu groß.

154 E Incorrect length in H format element.
Falsche Längenangabe im H-FORMAT-Element.

155 E Missing digits before §. Format element ignored.
Ziffernfolge vor X-, P- oder H-FORMAT-Element fehlt. Element wurde ignoriert.

156 S Format element beginning with § not recognizable.
FORMAT-Element beginnend mit "§" ist nicht deutbar.

157 E Repeat specification illegal in this context.
Einem Wiederholungsfaktor folgt ein unzulässiges FORMAT-Element.

158 S Minus not followed by P format element.
Nach '-' folgt kein P-FORMAT-Element.

159 S Format element § incomplete or zero.
Das FORMAT-Element "§" enthält unzulässige Ziffernfolge(n).

161 E Keyword specifier § ignored.
Der in der E/A-Anweisung angegebene Schlüsselwortparameter "§" wurde ignoriert.

162 E Wrong specifier §.
Fehler im Parameter "§".

163 S Specifier § missing or erroneous.
Parameter "§" fehlt oder hat falschen Typ.

164 E Missing comma inserted in data list.
In die Datenliste wurde ein fehlendes Komma eingefügt.

167 E No label in ERR or end-of-file specifier.
Der ERR- oder END-Parameter enthält keine Marke.

168 S UNIT=* not allowed in this statement.
UNIT=* ist in dieser Anweisung nicht erlaubt.

169 S Incompatible FMT and UNIT specifier.
Widersprüchliche UNIT- und FMT-Parameter.

172 S Error in element of data list.
Fehler in einem Datenlistenelement.

173 S List of data elements contains assumed size array.
Feld unbekannter Länge in der Datenliste.

179 S DO-variable of incorrect type.
Die implizite DO-Variable ist nicht von Typ INTEGER, REAL oder DOUBLE PRECISION.

180 S Expression in implied-do-control of incorrect type.
Ausdruck in der Steuerung für implizites DO ist nicht vom Typ INTEGER, REAL oder DOUBLE PRECISION.

181 S Terminal expression in implied-do missing.
In der Steuerung für implizites DO fehlt der zweite Ausdruck.

184 E Missing input/output list.
Auf Komma folgt keine Datenliste.

185 E Redundant commas ignored.
Überflüssige Kommas in der Datenliste wurden ignoriert.

186 E Unpermitted increment expression. 1 assumed.
Unzulässiger Ausdruck für Schrittweite. 1 wurde angenommen.

187 S REC specifier not allowed.
Unerlaubter Parameter REC für INTERNAL-FILE.

188 S Illegal format specifier for internal file.
LIST-gesteuerte oder unformatisierte E/A für interne Datei nicht erlaubt.

189 E Positional specifier not first of all. Ignored.
Stellungsparameter steht nicht am Listenanfang und wurde ignoriert.

190 E § illegal in this context.
"§" ist an dieser Stelle unzulässig.

201 S Symbolic name § illegal in this context.
Bezeichner "§" ist ungültig in Kontext.

202 S Symbolic name § neither in COMMON nor dummy argument.
Variable "§" muß außerhalb der Routine einen Wert erhalten.

203 S § illegal in left hand side of assignment.
Bezeichner "§" darf nicht auf der linken Seite der Ergibtanweisung auftreten.

204 S Array § must be a dummy argument.
Feld "§" muß ein formaler Parameter sein.

206 S Illegal use of dummy argument §.
Fehlerhafte Anwendung des formaler Parameters "§" der Anweisungsfunktion.

207 S Intrinsic name § used as actual argument.
INTRINSIC-Name "§" darf nicht als aktuelles Argument übergeben werden.

208 S § not allowed as actual argument.
Bezeichner "§" ist als aktuelles Argument ungültig.

209 S § not a function or subroutine name.
Bezeichner "§" ist kein Funktions- oder Unterprogrammname.

210 S § twice declared as dummy argument.
Der Bezeichner "§" wurde mehrfach als formaler Parameter einer Anweisungsfunktion vereinbart.

211 S § not allowed as dummy argument.
Bezeichner "§" darf nicht als formaler Parameter einer Anweisungsfunktion verwendet werden. Standarddatentyp wurde angenommen.

212 S § neither array nor statement function.
Bezeichner "§" auf der linken Seite der Ergibtanweisung ist kein Feld oder Bezeichner kann kein Anweisungsfunktionsname sein.

213 E § not of type INTEGER.
Bezeichner "§" hat nicht den Datentyp INTEGER.

214 S § not allowed in label assignment.
Bezeichner "§" darf nicht als Markenvariable verwendet werden. Der Standarddatentyp wurden angenommen.

215 S Multiple declared label.
Die Marke wurde mehrfach vereinbart.

216 S Substring § not of type CHARACTER.
Der als Teilkette verwendete Bezeichner "§" ist nicht vom Typ CHARACTER.

217 S § in illegal context.
Bezeichner "§" im Kontext nicht erlaubt..

218 S § in illegal context. May be consequence of error.
Bezeichner "§" im Kontext nicht erlaubt; möglicherweise Folgefehler.

222 E Control line ends statement. Control line ignored.
Innerhalb einer Anweisung tritt eine Steuerzeile auf. Die Anweisung wurde beendet und die Steuerzeile ignoriert.

240 S § may not be dummy argument.
Formaler Parameter "§" wurde bereits als SUBROUTINEN-Name oder formaler Parameter verwendet.

241 S Illegal dummy argument ignored.
Ungültiger formaler Parameter in der Liste der formalen Parameter wurde ignoriert.

242 E Incorrect length specification.
Ungültige Längenangabe in FUNCTION wurde ignoriert.

243 S § conflicts with use before.
Eintrittspunkt "§" steht im Widerspruch zur vorherigen Verwendung. Die Anweisung wurde ignoriert.

244 E Alternate return specifier near § ignored.
Alternative Rücksprünge in FUNCTION bei ENTRY "§" wurden ignoriert.

245 S § used before in COMMON, SAVE, or EQUIVALENCE.
Formaler Parameter "§" wurde bereits in COMMON, EQUIVALENCE oder SAVE verwendet.

246 S Implementation restriction. More than § dummy arguments.
Implementierungsbedingte Einschränkung. Die Parameterliste enthält mehr als "§" formale Parameter. Überzählige Argumente wurden ignoriert.

247 E Illegal praefix ignored.
Unerlaubtes Vorzeichen in der Längenangabe wurde ignoriert.

248 E Duplicate use of § in SAVE.
Zweimalige Verwendung des Elementes "§" in der SAVE-Anweisung. Das Element wurde ignoriert.

249 E § is INTRINSIC. Dummy argument ignored.
Formaler Parameter "§" wurde bereits als INTRINSIC-Funktion definiert. Er wurde ignoriert.

254 S Missing (near §.
Fehlende "(" vor Dimensionsliste bei "§".

255 S More than 7 dimensions.
Mehr als 7 Dimensionen angegeben.

256 E Invalid delimiter. Comma assumed.
Ungültiges Trennzeichen. Ein Komma wurde angenommen.

260 E Zero length specification. 1 assumed.
Längenangabe hat den Wert 0 oder ist nicht vom Typ INTEGER. 1 wurde angenommen.

262 E § not a parameter. 1 assumed.
"§" ist keine Konstante. 1 wurde angenommen.

263 S First * ends dimension list near §.
Der erste '*' beendet die Dimensionsliste von "§".

266 E Symbolic name expected.
Erwarteter symbolischer Name fehlt.

267 E Invalid length specification. 1 assumed.
Ungültige Längenangabe. 1 wurde angenommen.

268 E Invalid length specification. Default assumed.
Ungültige Längenangabe. Standardlänge wurde angenommen.

270 S § impossible as assumed size array.
Variable Dimensionsgrenzen bei "§", aber gleichzeitig in SAVE oder COMMON.

272 S § used before as global entity or in PARAMETER.
"§" ist bereits globale Größe oder symbolischer Name einer Konstanten.

273 S Invalid common name.
Fehlerhafter COMMON-Name.

275 S Assumed size.array § impossible in COMMON.
Feldgrenzen für "§" enthalten Variablen. Rest des COMMON-Bereichs wurde ignoriert.

276 S Type conflict near §.
COMMON-Element "§" widerspricht dem Datentyp der anderen COMMON-Elemente.

277 S Duplicate array declaration for §.
COMMON-Element "§" wurde bereits als Feld vereinbart.

278 S Common element expected or duplicate.
COMMOM-Element fehlt oder doppelt verwendet. Der Rest des COMMON-Bereiches wurde ignoriert.

280 S § in EQUIVALENCE with conflicting type.
"§" steht zu einer Variablen mit unverträglichen Datentyp in EQUIVALENCE.

282 E § redundant in EQUIVALENCE.
Redundantes Auftreten von "§" in EQUIVALENCE.

283 S Illegal substring or index specification for §.
Fehler in INDEX- oder Teilkettenangabe für "§".

284 S More than one element expected.
EQUIVALENCE-Gruppe enthält nur ein Element.

285 W Last index too large for §.
In der Indexliste von § ist der letzte Index zu groß.

286 S Index exceeds dimension bounds for §.
Index nicht in Dimensionsgrenzen bei "§".

287 S § not an array but used so.
"§" hat Indizes, ist aber kein Feld.

288 S Number of indexes conflicts with dimensions for §.
Indexanzahl für "§" stimmt nicht mit Dimensionsanzahl überein. Die Indexliste wurde ignoriert.

289 T Conflicting equivalence near §.
Unverträgliche Equivalence bei "§".

290 W Common § has no element.
COMMON-Bereich "§" hat kein Element.

291 T Common extented to left near § effected by EQUIVALENCE.
COMMON-Bereich wurde bei "§" durch eine EQUIVALENCE nach links erweitert.

292 T Common changed near § effected by EQUIVALENCE.
Abstand oder Reihenfolge der COMMON-Elemente wurde bei "§" durch EQUIVALENCE verändert.

293 S Invalid element in EQUIVALENCE.
Ungültiges EQUIVALENCE-Element.

295 E § can not be intrinsic.
"§" ist keine INTRINSIC-Funktion oder durch andere Verwendung zu Nicht-INTRINSIC geworden.

296 S § used as common name before.
Element "§" ist bereits ein COMMON-Name.

297 S Lower bound exceeds upper bound near §.
Untergrenze ist größer als Obergrenze im Felderklärer von "§".

300 E Text following an expression skipped. May be consequence of error.
Text zwischen Ausdruck und nächstem :,), Komma oder Anweisungsende wurde übergangen; oft Folgefehler eines Fehlers im Ausdruck.

301 S Symbolic name expected in input list.
Element in Eingabeliste ist kein Bezeichner.

302 S GT/GE/LT/LE-comparision with COMPLEX operand.
Ausdruck enthält GT/GE/LT/LE-Vergleich mit COMPLEX-Operanden.

303 S Expression type conflicts with operand type.
Operand paßt nicht zum verlangten Ausdruckstyp.

304 S § in constant expression misplaced.
Variable "§" tritt in einem Ausdruck auf, der konstant sein muß.

305 S Parameter § misplaced in input list.
Konstante "§" tritt in Eingabeliste auf.

306 S Text before) in substring list ignored.
Text vor ')' in Teilkettenliste wurde übergangen.

307 S Text before : in substring list ignored.
Text vor ':' in Teilkettenliste wurde übergangen.

308 S List of actual arguments for § ignored.
Parameterliste für Prozedur "§" muß leer sein. Die Parameter wurden ignoriert.

309 S List of actual arguments for § missing.
Parameterliste für Prozedur "§" darf nicht leer sein.

310 S Argument of § of wrong type.
Falscher Datentyp im aktuellen Parameter der INTRINSIC-Funktion "§".

311 S Misplaced .NOT. .
Operator .NOT. ist hier nicht erlaubt.

312 W Actual argument for § with variable length.
Für '(CHARACTER-OPERAND)' als aktuellen Parameter der Prozedur "§" kann keine temporäre Variable erzeugt werden, da Länge unbekannt ist. Der Operand wurde selbst als aktueller Parameter übergeben.

313 S Actual argument contains // with variable length operand.
Für einen CHARACTER-Ausdruck (mit //-Operator(en)) als aktuellen Parameter kann keine korrekte temporäre Variable erzeugt werden, da *-Längen aufgetreten sind. Der Parameter wurde wahrscheinlich mit falscher Länge übergeben.

314 S Text before ',' skipped in list of actual arguments near §.
In der Liste der aktuellen Parameter von Prozedur "§" wurde Text zwischen Parameter und ',' übergangen.

315 S Text before ')' skipped in list of actual arguments near §.
Schließende Klammer fehlt oder folgt nicht sofort auf Liste der aktuellen Parameter der Prozedur "§".

316 S Missing actual arguments in call of §.
Zu wenig aktuelle Parameter im Aufruf der SUBROUTINE "§".

317 S Too many actual arguments in call of §.
Zu viele aktuelle Parameter im Aufruf der SUBROUTINE "§".
Restliche Parameter wurden ignoriert.

318 S Redundant text in CALL statement.
Überflüssiger Text in CALL-Anweisung nach SUBROUTINEN-Aufruf.

319 S Wrong number of alternate returns.
Falsche Anzahl alternativer Ausgänge in der CALL-Anweisung.

320 E Wrong actual argument deleted.
Fehlerhafter aktueller Parameter wurde ignoriert.

321 E Invalid text ignored.
Nicht interpretierbarer Text in der Anweisung wurde übergangen.

322 E Length specification * incorrect. 1 assumed.
Längenangabe '*' ist nicht erlaubt. Länge 1 wurde angenommen.

323 E Duplicate use of letter.
Buchstabe darf nur einmmal in der IMPLICIT-Anweisung auftreten.

324 E Wrong expression type converted.
Ausdruck hat falschen Typ und wurde konvertiert.

325 S § invalid destination operand.
Unzulässiger Zieloperand "§" in der Ergibtanweisung.

326 S Missing = after destination operand.
'=' fehlt oder folgt nicht sofort auf Ziel.

327 S Conflict between source and destination. Assignment impossible.
Typ von Quelle und Ziel in der Ergibtanweisung unverträglich; Zuweisung nicht möglich.

328 S Conversion to LOGICAL impossible.
Keine Konvertierung zum Typ LOGICAL möglich.

329 S Redundant text at end of statement ignored.
Überflüssiger Text am Anweisungsende wurde ignoriert.

330 S Empty argument in list of actual arguments.
'(,' oder ',,' oder ',)' in der Liste der aktuellen Parameter.

331 E Intervall specification invalid.
Buchstabenbereich nicht in alphabetischer Folge.

332 S Expression not of type LOGICAL.
Ausdruck hinter IF/ELSEIF ist nicht vom Typ LOGICAL.

333 S ENDIF inserted.
ENDIF wurde vor END-Anweisung eingefügt, da ENDIF fehlt oder in DO-Gruppe auftritt.

334 S Invalid length of substring §. 1 assumed.
Fehler in Teilkettengrenzen für Variable "§".

335 S Invalid reference of function §.
Unzulässiger Aufruf der CHARACTER-Funktion "§". Bei Aufruf darf die Länge des Funktionswertes nicht als *-Länge vereinbart sein.

336 S Expression of wrong type. Conversion impossible.
Ausdruck hat falschen Typ, Konvertierung nicht möglich.

337 W Critical conversion in assignment.
Bedenkliche Konvertierung bei Zuweisung.

338 E Integer constant too large in assignment.
INTEGER-Konstante zu groß bei Zuweisung.

402 S Unable to evaluate a constant expression.
Die Berechnung eines Konstantenausdrucks führt zu einer Unterbrechungsbedingung.

407 E Name list contains illegal entity.
Unzulässiges Element in der Namensliste.

408 E Initialize of § prohibited.
Initialisierung von "§" ist nicht erlaubt.

413 S Implementation restriction. Implied-DO nesting too deep.
Implementierungsbedingte Einschränkung. IMPLIED-DO-Liste zu tief geschachtelt.

414 E Implied-DO list contains illegal entity.
Ungültiges Element in der DO-Liste.

416 S Implementation restriction. INTEGER constant exceeds 32767.
Implementierungsbedingte Einschränkung. Indexliste enthält Wert > 32767.

419 E Illegal entity in index list for §.
Indexliste von "§" enthält ungültiges Element.

420 E Illegal use of § in name list .
"§" ist ungültiges Element der Namensliste.

422 E § not a parameter or invalid.
"§" ist kein Konstantenname.

423 E Type of § conflicts with type of initial value.
Datentyp von "§" ist nicht mit dem Typ des zugehörigen Anfangswertes verträglich.

424 E Repeat specification not of type INTEGER.
Wiederholungsfaktor ist keine INTEGER-Konstante.

426 E Data list ended before name list.
Datenliste abgearbeitet, Namenliste enthält noch Elemente.

427 E § not of type INTEGER.
"§" in Index-/Laufvariablenausdruck ist keine INTEGER-Konstante.

428 E Data list missing. May be consequence of error.
Datenliste fehlt.

432 S Praefix operator conflicts with type of operand.
Präfixoperator widerspricht dem Datentyp des Operanden.

433 E Erroneous repeat specification.
Fehlerhafte Wiederholungsspezifikation.

434 E Illegal text after data list.
Ungültiger Text nach Datenliste.

435 E Data list without closing /.
Datenliste nicht durch "/" beendet.

437 S Invalid type of DO variable.
Falscher Datentyp für Laufvariable "§" der DO-Anweisung, Anweisung wurde ignoriert.

439 S Expression starts with two or more unary operators.
Ausdruck beginnt mit zwei oder mehr unären Operatoren.

440 S End of DO group missing.
Offene DO-Gruppen am Ende der Programmeinheit, DO-Gruppen wurden vor der END-Anweisung beendet.

441 S DO group exceeds end of IF/ELSEIF block.
DO-Gruppe erstreckt sich über das Ende eines IF/ELSEIF-Blocks, DO-Gruppe wurde vor der ELSEIF-, ELSE- oder ENDIF-Anweisung beendet.

442 S Statement not permitted. Replaced by CONTINUE.
Anweisung ist in einer DO-Gruppe nicht erlaubt und wurde durch CONTINUE ersetzt.

443 S Overlapping DO group.
Verbotene Überlappung von DO-Gruppen, für den Abschluß der inneren DO-Gruppe wurde Code eingefügt.

444 S Erroneous end of DO group. Replaced by CONTINUE.
Die Anweisung ist nicht als Ende einer DO-Gruppe erlaubt und wurde durch CONTINUE ersetzt.

445 S Comma missing after expression.
',' folgt nicht oder nicht sofort auf Ausdruck in der DO-Anweisung, oft Folge eines Fehlers im Ausdruck.

447 S Incrementation expression erroneous. 1 assumed.
Ausdruck für Schrittweite in DO-Anweisung unvollständig, 1 wurde angenommen.

448 T Implementation restriction. DO nesting too deep.
Implementierungsbedingte Einschränkung. Zu tiefe Schachtelung von DO-Gruppen.

450 S Invalid label reference.
Marke in DO-Anweisung bezieht sich auf frühere Anweisung, DO-Anweisung wurde ignoriert.

452 S Text ignored between expression and label.
Überflüssiger Text zwischen arithmetischen Ausdruck und Marken wurde ignoriert, oft Folge von Fehler im Ausdruck.

453 S Erroneous label §.
Fehler in Marke "§" der arithmetischen IF-Anweisung.

454 S Expression not of type LOGICAL. .TRUE. assumed.
Ausdruck in logischer IF-Anweisung ist kein logischer Ausdruck, .TRUE. wurde angenommen.

455 S Statement missing after expression.
Bedingte Anweisung fehlt in logischer IF-Anweisung, CONTINUE wurde angenommen.

456 S Text ignored between IF expression and statement.
Überflüssiger Text zwischen logischem Ausdruck und Schlüsselwort wurde ignoriert.

457 S Statement invalid after IF expression.
Anweisung in logischer IF-Anweisung ist dort verboten, nicht erkennbar oder falsch und wurde durch CONTINUE ersetzt.

459 S Erroneous label after GOTO.
Fehler in Marke der unbedingten GOTO-Anweisung nach logischen IF, GOTO wurde durch CONTINUE ersetzt.

460 E Incrementation value of 0 replaced by 1.
Schrittweite 0 wurde durch 1 ersetzt.

461 S Implementation restriction. Length exceeds 32767 near §.
"§" hat Längen größer 32767. 1 wurde angenommen.

462 S Implementation restriction. Offset exceeds 32767 near §.
"§" hat Index- oder Teilkettenangabe größer 32767.

463 S Implementation restriction. Equivalence offset exceeds 32767 near §.
EQU-Offset für "§" größer 32767.

470 S Incorrect use of label §.
Sprung zu MARKE "§" ist unzulässig. Einsprung in einen DO- oder IF-Block.

471 S Incorrect use of label §.
Marke "§" ist an einer FORMAT-Anweisung definiert. Verzweigung ist nicht möglich.

472 S Use of label § in I/O list prohibited.
Marke "§" ist nicht an einer FORMAT-Anweisung definiert. Verwendung in E/A-Liste ist unzulässig.

473 S Incorrect use of label §.
Marke "§" ist an einer ELSEIF-, ELSE- oder ENDIF-Anweisung definiert.

474 S Incorrect use of label §.
Marke "§" ist an einer nicht ausführbaren Anweisung definiert.

475 S Undefined label §.
Marke "§" ist nicht vereinbart.

476 E Label § shortened to 5 digits.
Die Marke "§" hat einen unerlaubten Wert. Die Marke wurde gekürzt.

477 T Implementation restriction. More than 99 statement functions.
Implementierungsbedingte Einschränkung. Die Programmeinheit enthält mehr als 99 Anweisungsfunktionen.

480 T Implementation restriction. Data segment overflow.
Implementierungsbedingte Einschränkung. Gesamtlänge aller Variablen im DATA-Segment größer als 65535.

481 T Implementation restriction. Common § overflow.
Implementierungsbedingte Einschränkung. Gesamtlänge aller Variablen im COMMON-Segment "§" größer als 65535.

483 T Unable to open A86-file.
Die A86-Datei kann nicht erfolgreich eröffnet werden.

485 T Implementation restriction. Temporary storage overflow.
Implementierungsbedingte Einschränkung. Länge oder Anzahl der temporären Variablen ist zu groß.

486 T Implementation restriction. Operand stack overflow.
Implementierungsbedingte Einschränkung.
Operandenstapel-Überlauf. Anweisung ist zu kompliziert.

Anlage 3 Fehlercodes der Ausführungszeitfehler

0002H zu wenig Speicherplatz, Programm zu groß
0003H Kein gültiges Segment
0005H Zu viele temporäre Dateien
0010H READ nach EOF
0020H Datei existiert
0021H Datei existiert nicht
0023H Funktion wird vom System nicht unterstätzt.
0025H Keine Daten in der Datei
0030H Falscher Modus
0031H READ für leeren Block
0032H Kein freier Datenblock
0033H Alter Bereich kann vor READ für Block im neuen Bereich nicht abgeschlossen werden.
0034H Zugriff auf nicht vorhandenen Bereich
0035H Kein Platz für Verzeichniseintragung
0036H Satznummer außerhalb des gültigen Bereichs
0070H Eingabebereich zu kurz
11XXH mit UNIT=XX verbundene Datei kann nicht abgeschlossen werden
1200H ungültige STATUS-Angabe in OPEN-Anweisung
1201H ungültige ACCESS-Angabe in OPEN-Anweisung
1202H ungültige FORM-Angabe in OPEN-Anweisung
1203H ungültige BLANK-Angabe in OPEN-Anweisung
1204H ungültige STATUS-Angabe in CLOSE-Anweisung
1210H STATUS=NEW in OPEN-Anweisung für existierende Datei nicht erlaubt
1211H OPEN-Anweisung mit FILE-Angabe verlangt STATUS=NEW oder STATUS=OLD
1212H STATUS=KEEP für SCRATCH-Datei nicht erlaubt
1215H STATUS=OLD für nicht existierende Datei nicht erlaubt.
1220H ACCESS in OPEN-Anweisung gibt für die Datei nicht erlaubten Zugriff an
1221H FORM in der OPEN-Anweisung gibt Format an, das dem Dateiformat widerspricht
1222H BLANK in OPEN-Anweisung nicht erlaubt, wenn FORM=UNFORMATTED angegeben wird
1230H Für eine Direktzugriffsdatei muß in der OPEN-Anweisung RECL-Angabe vorhanden sein
1232H Auf die Datei kann nur sequentiell zugegriffen werden
1233H Die RECL-Angabe in der OPEN-Anweisung steht im Widerspruch zur Satzlänge der Datei
1234H Die geforderte Satzlänge für die Datei ist größer als 32767
1240H Fehler beim Eröffnen der Datei
1250H BACKSPACE, ENDFILE oder REWIND für die Datei nicht erlaubt (kein sequentieller Zugriff oder Standard-E/A-Datei)
1251H BACKSPACE für nicht-formatisierte Datei ohne RECL-Angabe nicht erlaubt
1252H kein Rückwärtslesen der Datei für BACKSPACE oder REWIND möglich
1253H BACKSPACE für nicht-existierende Datei nicht erlaubt
1281H E/A-Anweisung schon aktiv
1282H implementierungsbedingte Einschränkung: UNIT-Nr. zu groß
1283H implementierungsbedingte Einschränkung: zu viele Dateiverbindungen aktiv
1284H implementierungsbedingte Einschränkung: zu tiefe Schachtelung von Formatlisten

1285H	Dateiende
1287H	'UNIT' nicht angegeben
1289H	keine Dateivorverbindung
1290H	keine Dateiverbindung für 'DIRECT'
1291H	Unverträglichkeiten zwischen E/A-Anweisung und Dateiverbindung
1295H	Satznummer ist negativ
1296H	Satz existiert nicht
12A0H	zu viele Formatelemente für einen Satz
12A1H	falsche Satzlänge
12A2H	aktuelle Position im Satz falsch
12A5H	Interne Datei hat zu wenig Sätze
12B0H	falscher Datentyp vom Compiler geliefert
12B1H	Daten übersteigen Satzlänge
12B2H	Satz- und Dateilänge unverträglich
12C0H	zum Datentyp nicht passende Datendarstellung
12C1H	zu Formatelement Iw nicht passendes Datenelement
12C2H	zu Formatelement Iwm nicht passendes Datenelement
12C3H	zu Formatelement Fwd nicht passendes Datenelement
12C4H	zu Formatelement Ewd nicht passendes Datenelement
12C5H	zu Formatelement EwdEe nicht passendes Datenelement
12C6H	zu Formatelement Dwd nicht passendes Datenelement
12C7H	zu Formatelement Gwd nicht passendes Datenelement
12C8H	zu Formatelement GwdEe nicht passendes Datenelement
12C9H	zu Formatelement Lw nicht passendes Datenelement
12CAH	zu Formatelement A nicht passendes Datenelement
12CBH	zu Formatelement Aw nicht passendes Datenelement
12E1H	Formatliste enthält kein wiederholbares Formatelement
12E2H	Zeichenketten-Formatliste fehlerhaft
12E3H	Formatelement verfälscht
12E6H	Format-Angabe zu lang
12E8H	falsches Zeichen im Eingabebereich
12E9H	Gleitkomma-Überlauf bei Konvertierung
12EAH	INTEGER zu groß
1501H	Fehler bei der Dateivorverbindung
1601H	Kein ASSIGN für Markenvariable
1602H	Marke nicht in Markenliste
8020H	Division einer komplexen Zahl durch (0,0) aufgetreten
8021H	CLOG für (0,0) aufgetreten
8022H	(0,0) ** i aufgetreten für i < 0
8023H	(0,0) ** c und Realteil von c ≤ 0 oder Imaginärteil von c ≠ 0

Anlage 4 Modul- und PUBLIC-Namen der Laufzeitbibliotheken des FOR77

Bibliothek	Modulname	PUBLIC-Name
F77INI.L86	ASSIGNED_GOTO	FQ_GO_AL, FQ_GO_AS
	CHANGE_INTEGER_SIGN	TQ_160, TQ_170, TQ_171, TQ_190, TQ_191
	CMP_DIM_MIN_MAX	TQ_150, TQ_152, TQ_180, TQ_181, TQ_200, TQ_210
	COM2II	FQ_COM2I
	COMI2I	FQ_COMI2
	COMIII	FQ_COMII
	COMPUTED_GOTO	FQ_860, FQ_861
	DIVI	FQ_DIVI
	EXPIII	FQ_EXPII
	F77INI	F77_INI, F77_PC
	F77_INTER	FI_INTER
	FCHCOMP	FI_IGE, FI_IGT, FI_LLE, FI_LLT, F_CHCOMP
	FINDEX	FI_INDEX
	FI_CHAR	FI2_CHAR, FI_CHAR
	FQMD	FQMD
	FQ_CFBL	FQ_CFBL1, FQ_CFBL2, FQ_CFBL3, FQ_CFBL4, FQ_CFBL5, FQ_CFBL6, FQ_CFBL7
	FQ_EXT	FQ_EXT
	FQ_INT	FQ_INT
	FQ_PAUSE_STOP	FQ_911, FQ_915
	FQ_PGM	FQ_PGM, PGM_ENDE
	FQ_STOP	FQ_STOP
	F_MOVE	F_MOVEC, F_MOVEL
	ICHECK	UQ_100
	IDIV32	TQ_110
	IEXP	TQ_140, TQ_141
	IMOD	TQ_120, TQ_121
	IMUL32	TQ_100, TQ_102
	INDCHK	FQ_IND2, FQ_IND
	LABSI	FI_IABS
	LDIMI	FI_IDIM
	LENI	FI_LEN
	LMODI	FI_MOD
	LSIGNI	FI_ISIGN
	MAXI	MQ_MAXI
	MAXRL	MQ_MAXRL
	MAXL	MQ_MAXL
	MINI	MQ_MINI
	MINL	MQ_MINL
	MINRL	MQ_MINRL
	MULI	FQ_MULI
	SABSI	FI_2ABS
	SDIMI	FI_2DIM
	SMODI	FI_2MOD
	SSIGNI	FI_2SIGN
	TQESTART	TQESTART

Bibliothek	Modulname	PUBLIC-Name
F77INI.L86	TQFLE	TQ_IN_8087_ERH
	TQGETERH	TQGETERH
	TQWHERESTRAP87	TQWHERESTRAP87
	TQ_CI4	TQ_CI4
	TQ_FLPERH	TQ_FLPERH
	TQ_HEXOUT	TQ_HEXOUT
	TQ_LRSERH	TQ_LRSERH
	TQ_MSGOUT	TQ_MSGOUT
	TQ_PUSHSTR	TQ_PUSHSTR
	TQ_SYSERH	TQ_SYSERH
	TQ_TRAP87	TQ_TRAP87
	VCHECK	FQ_VCHECK
F77ART.L86	ABSI	FI_ABS
	ACOSI	FI_ACOS
	ACS	MQERACS, TAB_3
	AIMAGI	FI_AIMAG
	AINT	MQERAIN, MQERANT, MQERIAX, MQERICX, MQERIEX, MQERRNT
	AINTI	FI_AINT
	ALOG1I	FI_ALOG1
	ALOGI	FI_ALOG
	AMOD	MQERMOD, MQERRMD
	AMODI	FI_AMOD
	ANINTI	FI_ANINT
	ASINI	FI_ASIN
	ASN	MQERASN, TAB_4
	AT2	MQ_AT2
	ATAN	MQERATN, TAB_12
	ATAN2	MQERAT2, MQERATC, TAB_13
	ATAN2I	FI_ATAN2
	ATANI	FI_ATAN
	CABSI	FI_CABS
	COMMON_SIN_COS	MQ_COS, MQ_SIN
	CONI	FI_CON, FI_CON2C, FI_CONDC, FI_CONIC, FI_CONSC
	CONJGI	FI_CONJG
	CONSTANTS	MQ_CONST
	COS	MQERCOS, TAB_2
	COSH	MQERCSH, TAB_8
	COSHI	FI_COSH
	COSI	FI_COS
	CP2N63	MQ_CP2N63
	DABSI	FI_DABS
	DACOSI	FI_DACOS
	DASINI	FI_DASIN
	DATA2I	FI_DATA2
	DATANI	FI_DATAN
	DCOSHI	FI_DCOSH
	DCOSI	FI_DCOS
	DDIMI	FI_DDIM
	DECIDE	MQ_DECIDE
	DEXPI	FI_DEXP
	DIM	MQERDIM
	DIMI	FI_DIM
	DINTI	FI_DINT

Bibliothek	Modulname	PUBLIC-Name
F77ART.L86	DLOG1I	FI_DLOG1
	DLOGI	FI_DLOG
	DMODI	FI_DMOD
	DNINTI	FI_DNINT
	DPROD	FI_DPROD
	DSIGNI	FI_DSIGN
	DSINHI	FI_DSINH
	DSINI	FI_DSIN
	DSQRTI	FI_DSQRT
	DTANHI	FI_DTANH
	DTANI	FI_DTAN
	EXP	MQEREXP, TAB_5
	EXPI	FI_EXP
	EXPR2I	FQ_EXPR2
	EXPRII	FQ_EXPRI
	EXPRRI	FQ_EXPRR
	F77_INTER	FI_INTER
	IDNINI	FI_IDNIN
	INT	MQERIA4, MQERIC4, MQERIE4, MQERINT, MQERIRT, MQERNIN
	IRCHK	MQ_IRCHK
	LN	MQERLGE, TAB_11
	LOG10	MQERLGD, TAB_10
	MAX	MQERMAX
	MAXD	MQ_MAXD
	MAXLR	MQ_MAXLR
	MAXR	MQ_MAXR
	MIN	MQERMIN
	MIND	MQ_MIND
	MINLR	MQ_MINLR
	MINR	MQ_MINR
	MQRPI	MQ_MQRPI
	MQ_1	MQ_1
	MQ_63U	MQ_63U
	MQ_63U1	MQ_63U1
	MQ_63UPI2	MQ_63UPI2
	MQ_CCOS	MQ_CCOS
	MQ_CEXP	MQ_CEXP
	MQ_CLOG	MQ_CLOG
	MQ_CSIN	MQ_CSIN
	MQ_CSQRT	MQ_CSQRT
	MQ_EXM1	MQ_2XM1, MQ_EXM1
	MQ_I	MQ_I
	MQ_LOG	MQ_LOG, MQ_LOG10, MQ_LOGDN
	MQ_NAN	MQ_NAN
	MQ_NOF	MQ_NOF
	MQ_NQ	MQ_NQ
	MQ_OF	MQ_OF
	MQ_P0	MQ_P0
	MQ_PI2	MQ_PI2
	MQ_PII	MQ_PII
	MQ_Q	MQ_Q
	MQ_TLOG	MQ_TLOG
	MQ_U0	MQ_U0
	NINTI	FI_NINT
	NORM	MQ_NORM

Bibliothek	Modulname	PUBLIC-Name
F77ART.L86	RAD	MQ_RAD
	RERR	MQ_EXIT, MQ_RERR
	RI2	MQERCI2, MQERIA2, MQERIC2, MQERIE2, MQERNI2, MQERRI2
	ROUND	FQ_ROUND
	SIGN	MQERSGN
	SIGNI	FI_SIGN
	SIN	MQERSIN, TAB_7
	SINH	MQERSNH, TAB_1
	SINHI	FI_SINH
	SINI	FI_SIN
	SQRT	MQ_SQRT
	SQRTI	FI_SQRT
	TAN	MQERTAN, TAB_6
	TANH	MQERTNH, TAB_9
	TANHI	FI_TANH
	TANI	FI_TAN
	TQ_CC	TQ_CC
	TQ_CI2	TQ_CI2
	TQ_DIVC	TQ_DIVC
	TQ_MULC	TQ_MULC
	TXAM	MQ_TXAM
	Y2X	MQERY2X, MQERYI2, MQERYI4, MQERYIS, TAB_14
	YL2X	MQ_YL2X
F77EAS.L86	FEA_DIRECT_SEEK	FEA_SEEK
	FEA_FE_CTLT	FE_CL_BRACKET, FE_OP_BRACKET, FE_REPEAT
	FEA_FE_CTL2	FE_BLANK, FE_SCALE, FE_S
	FEA_FE_CTL3	FE_T, FE_X
	FEA_FE_CTL4	FE_ZKONST
	FEA_FE_CTL5	FE_SLASH
	FEA_FMTD_DATA	DATEN_TYP, FEA_FMTD_DATA
	FEA_FMTE_FORMAT	FORMAT_TYP
	FEA_FORMAT_CONTROL	FEA_FMT_CTL_E, FEA_FMT_CTL_8 FEA_FMT_CTL_I
	FEA_FORMAT_EXTINT	M_FEI198, FEA_FMT_EXTINT
	FEA_FORMAT_EXTINT2	ZK_ZIFFER1, ZK_NEXT, M_EI2600, M_EI2500, ZK_INCORRECT, FEA_FMT_EXTINT2
	FEA_FORMAT_EXTINT3	ZK_T, ZK_S, ZK_Q, ZK_L, ZK_I, ZK_G, ZK_F, ZK_E, ZK_D, ZK_B, ZK_A, ZK_COLON, ZK_ZIFFER, ZK_SLASH, ZK_MINUS, ZK_CL_PAR, ZK_OP_PAR, ZK_APO
	FEA_FORMAT_SEEK	FEA_FFSINIT, FEA_FORMAT_SEEK
	FEA_HELP	CONN_OK, FREES_FDB, ALLOC_FDB, UNIT_SEEK
	FEA_IMPLICIT_OPEN	FEA_IMP_OPEN

Bibliothek	Modulname	PUBLIC-Name
F77EAS.L86	FEA_INTERNAL_FILE	FEA_INT_FILE
	FEA_LIST_FORMAT_INPUT	FEA_LISTF_DATA
	FEA_LIST_FORMAT_OUTPUT	FEA_LISTU_DATA
	FEA_PRECON_SEEK	M_PRECON
	FEA_PROMPT	FEA_PROMPT
	FEA_READ_CTL	FEA_READ
	FEA_RECORD_ANZAHL	FEA_RECANZ
	FEA_TYPE_LEN_TABLE	TYP_TABLE
	FEA_UNFORMAT_CONTROL	FEA_UFD_FILL
	FEA_WRITE_CTL	FEA_WRITE
	FEA_ZK_VGL	ZK_VGL
	FFS_A_FORMAT	FE_O_CHAR, FE_I_CHAR
	FFS_CWD_CWDEE_OUTPUT	FFS_GWDEE, FFS_GWD
	FFS_EWD_DWD_EWDEE_OUTPUT	FFS_EWDEE, FFS_DWD, FFS_EWD
	FFS_FWD_OUTPUT	FFS_FWD
	FFS_I_FORMAT	FE_O_IWM, FE_O_IW, FE_I_INTEG
	FFS_L_FORMAT	FE_O_LW, FE_I_LW
	FFS_Q_FORMAT	FE_O_Q, FE_I_Q
	FFS_REAL_INPUT	FE_I_REAL
	FFS_REAL_OUTPUT	FE_O_GWDEE, FE_O_GWD, FE_O_DWD, FE_O_EWDEE, FE_O_EWD, FE_O_FWD
	FFS_SERVICE_T1	FFSINTEG, FFSDECIM
	FFS_SERVICE_T2	FFSREALK
	FFS_SERVICE_T3	FFSDZAHL
	FFS_SERVICE_T4	FFSRMANT
	FFS_SERVICE_T5	FFSROUND
	FFS_SERVICE_T6	FFSREALI
	F77_ACS_ATTACH_CREATE	DQATTACH, DQCREATE
	F77_ACS_ALLOCATE	DQALLOCATE
	F77_ACS_CLOSE	DQCLOSE
	F77_ACS_DELETE	DQDELETE
	F77_ACS_DETACH	DQDETACH
	F77_ACS_FREE	DQFREE
	F77_ACS_GET_CONNECTION_STATUS	DQGETCONNECTIONSTATUS
	F77_ACS_OPEN	DQOPEN
	F77_ACS_READ	DQREAD
	F77_ACS_SEEK	DQSEEK
	F77_ACS_TRUNCATE	DQTRUNCATE
	F77_ACS_WRITE	DQWRITE
	F77_AUX	F77_AUX_STMTS
	F77_BER1	F77_BSEFRW
	F77_BER2	M_BER_200, M_BER_120, F77_BSEFRW2
	F77_CLOSE_FILE	F77_CLOSE_ALL, F77_CLOSE

Bibliothek	Modulname	PUBLIC-Name
F77EAS.L86	F77_EA_CONTROL	FEAROUT1, A_FILE_NAME, OFFS_DLE, A_SCHALTER, B_SCHALTER, CURRENT_UNIT, CURRENT_FILE, ALL_FSB_ENDE, ALL_FSB, ALL_FDB_ENDE, ALL_FDB, LAST_FSB, FIRST_FSB, FREE_FSB, LAST_FDB, FIRST_FDB, FREE_FDB, FDB$UNIT0, RET_SP, RET_DI
	F77_EA_ENDE	M_EAR900, M_EAR104, M_EAR100
	F77_FQCLOSE	CLO_MRC, FQ_CLOSE
	F77_INQ1	F77_INQUIRE
	F77_INQ2	M_INQ_300, M_INQ_200, M_INQ_140
	F77_OPEN1_FILE	PRE_WERT, F77_OPEN
	F77_OPEN2_FILE	ACCESS_BYTE, M_OPE_070, F77_OPEN2
	F77_OPEN3_FILE	M_OPE_400, M_OPE_300, F77_OPEN3
	F77_PRELIST	F77_PRELIST

Anlage 5 Implementierungsbedingte Besonderheiten

- FOR77 verwendet den ASCII-Code.
- Felder können maximal einen Speicherplatz von 32767 Bytes belegen.
- Markenlisten in der GOTO-Anweisung können maximal 50 Marken enthalten.
- Die maximale Schachtelungstiefe von DO-Anweisungen ist 25.
- Die Länge eines COMMON-Bereiches darf 65535 nicht übersteigen.
- Die Summe der Längen aller Elemente im DATA-Segment darf 65535 nicht übersteigen.
- Eine Programmeinheit darf nicht mehr als 99 Anweisungsfunktionen enthalten.
- Die Ausführung einer PAUSE-Anweisung führt zur Terminal-Meldung

  PROGRAM PAUSE text

  Die Programmausführung wird erst fortgesetzt, wenn der Anwender eine beliebige Taste betätigt. Die Eingabe von CTRL-C oder "S" bewirkt die Beendigung des Programms mit impliziten Abschließen aller Dateien.
- Die Ausführung einer STOP-Anweisung führt nach der Terminal-Meldung

  PROGRAM STOP text

  zur Programmbeendigung.
- Bei der Überlagerung mit EQUIVALENCE dürfen keine Offsets größer 32767 entstehen.
- Zeichenkettenvariablen dürfen nicht länger als 32767 sein.
- Die Gesamtlänge des CODE-Segment (ausführbare Anweisungen einer Programmeinheit) ist auf 65535 beschränkt.
- Zur Ausführungszeit können maximal 6 Dateien gleichzeitig eröffnet sein.
- Bei Überlagerungen mit EQUIVALENCE in einem COMMON-Bereich muß die Anfangsadresse des überlagerten COMMON-Elementes in den ersten 32K des COMMON-Bereichs liegen.
- Die Anzahl der formalen Parameter in einer ENTRY-Anweisung ist auf 30 beschränkt.

Literaturverzeichnis

[1] C 1016-0200-1 FORTRAN77 - Sprachbeschreibung

[2] C 1014-0003-1 RASM86 - Benutzungshinweise in "Programmpaket für modulare Programmierung"

[3] C 1014-0003-1 RASM86 - Sprachbeschreibung in "Programmpaket für modulare Programmierung"

[4] C 1014-0003-1 LINK86 - Benutzungshinweise in "Programmpaket für modulare Programmierung"

[5] C 1014-0003-1 LIB86 - Benutzungshinweise in "Programmpaket für modulare Programmierung"

[6] C 1014-0001-1 Numerische Bibliotheken

Sachwortverzeichnis

A-Option 15
Abbruch der Übersetzung 13
Abbruchfehler 41
Abkürzungen 7
Abschließendes Leerzeichen 31
ACCESS-Parameter 30
Adresse der Argumentliste 26
Aktive Dateiverbindung 29
AL-Option 16
ALIST 12, 16
allgemeine Fehler 42
Anschluß von Assembler-Programmen 25
Anschlußbedingungen 25
Anweisungsnummer 19
Arbeitsdatei 10
ASCII-Code 48
Assemblerprogramm 8, 9
Assemblerquelltext 11
Auflistungen
    auf dem Terminal 18
    des FOR77 18
    in der PRN-Datei 19
Aufruf
    des FOR77 10
    des Linkers LINK86 24
    eines FORTRAN77-Programms 27
    von RASM86 22
Ausdrucksoptimierung 46
Ausführen von FORTRAN77-Programmen 27
Ausgabedateien des Compilers 12
Ausnahmebedingungen des Gleitkomma-Emulators 36
A86 11, 20

Bereichsgrenzen der Datentypen 48
Bias 33
Bibliothekar LIB86 24
Bibliotheken 24
Bibliotheken zur Programmverbindung 23

C-Option 15
CHARACTER-Daten 48
Code-Segment 21
COM-Option 16
COMMON-Bereich 20
COMPILE 13
Compiler 8
CON 28
CPASS 9
CR 7, 29
CTRL-Z 7

D-Option 15
Datei
    - physische 31
    - temporäre 31
Dateispezifikation 10, 18, 23, 27

Dateityp 10
- CMD 24
- L86 24
- OBJ 24
Dateityp A86 9
Dateivorverbindung 27
Dateizugriffsarten 30
Daten-Segment 20
DB-Option 16
DEBUG 12
DEBUG-Möglichkeiten 9
denormalisierter Wert 36
DENORMALIZED OPERAND 38
DIRECT 30
Direktzugriff 30
DPASS 9
Drucker 27

E/A
-Routinen 23
-Steuersystem 21
E-Option 15
EANULL.L86 23
effektive Programmierung 47
EJ-Option 16
EJECT 13
EL-Option 16
EOF 32
EPASS 9
ERR 11
ERR-Datei 14
ERRORLIMIT 13
ESEG-Direktive 20
EWM87.L86 24
EWM87,OBJ 24
Exponent 33
Extra-Segment 20
EXTRN-Direktive 20

F-Option 15
Fehleranzahl 13
Fehlerbehandlungssystem 21
Fehlercode 32
Fehlercodes 64
Fehlerdatei 40
Fehlermeldungen
- E/A 43
- zur Übersetzungszeit 40, 49
- zur Ausführungszeit 42
Fehlermeldungsliste 9
Fehlernummer 40
Fehlerprotokoll 11
Fehlerschwere 13, 41
Feldbeschreiber 20
Festkomma-Überlauf 44
Festkomma-Nulldivision 44
FF-Option 16
FIPASS 9
Formatelemente 31
formatisierte Datei 31

Fortsetzungszeile 14
FOR77 8, 9
FREEFORM 14
freies Format 11
Funktionswert 26
- Rückgabe 26
F77ART.L86 24
F77EAS.L86 23
F77INI.L86 24

G-Option 15
Geräteangabe 10
Gültigkeitsbereich 33
Gleitkomma
- Arithmetik 33
- Bibliothek 21
- Datentyp 23
- Emulator 23, 33, 36
- Fehler 44
- Routinen 23
- Zwischenresultate 35
Globale Prüfung 9
GOSTATEMENT 12
GS-Option 16
GXPASS 9

Hauptspeichergröße 8

I-Option 9, 15
INC-Option 16
INCLUDE-Option 14
Information 41
INITFP 21
Initialisierung 9
INTEGER 43
- OVERFLOW 44
- Variable 13
- ZERO-DIVIDE 44
Interne Darstellung 48
INVALID OPERATION 36
IOSTAT-Parameter 32
IOSTAT-Werte 32

Kommandozeile 10
Kommentarzeile 12
Kopfzeile 18
KPASS 14

L-Option 16
Laufzeitbibliothek 8
Laufzeitinitialisierung 35
Leerzeichen 11
Lexikalische Analyse 9
LF 7, 29
LIB86 8
LINK86 8, 24
LIST 16
LIST-Direktive 12
Listendatei 16
Lochbandleser 27

Lochbandstanzer 27
LOGICAL-Daten 48
LOGICAL-Variablen 13
LST 27

Mantisse 33
Masken für Ausnahmebedingungen 33
Modulnamen 66

Name des Laufwerkes 18
NAME-Direktive 20
NOCOMPILE 13
normalisierter Wert 36
Notation 7
NUL 27

OPASS 9
OPT 14, 16
Optimierung 46
Optimierung bei DO-Schleifen 46
Optimierungsstufe 46
Optionen
    der Kommandozeile 10, 15
    der Steuerzeile 16
    des FOR77 12
OVERFLOW 39

P-Option 15
Pässe 9
PAGELENGTH 12
PAGEWIDTH 12
PAUSE 72
Phase 9
PL-Option 16
Position des impliziten Kommas 33
PRECISION 33, 39
PRN 11
PRN-Datei 12
Programmvariable 20
Pseudo-Datei 27
Pseudoroutinen 23
PUBLIC-Namen 66
PUN 27
PW-Option 16

Q-Option 15
Quelltextauflistung 19
QUIET 14

RANGE 44
RASM86 8, 22
RDR 27
REAL-Variablen 13
REC-Parameter 30
RECL-Parameter 29
Registerzuordnung 9
RETF 26
Rückkehrcode 34
RPASS 9

Rundungsmodus 35

S-Option 11, 15
Satz
- Länge 29, 30
- relative Adresse 31
- Struktur 29, 30
- Trennzeichen 29, 30
SB-Option 16
SEARCH 25
Seitenvorschub 12
SEQUENTIAL 30
SPASS 9
SPILLFIL 11
STACK-Größe 21
Stack-Segment 21
Stack-Verkürzung 25
Standard-Vorverbindungen 28
Standardformat 14
Statuswort (SW) 33, 34
Steuersystem für Ein- und Ausgabe 21
Steuerwort (CW) 33
STG-Option 16
STOP 72
STORAGE 13
Struktur des Compilers 9
STT-Option 16
SUBTITLE 13
SYMBOLS 12
Symboltabelle 12
Syntaktische Analyse 9
S1PASS 9
S2PASS 9

T-Option 11, 15
TEMPREAL
- Argument 33
- Format 33
Terminal 27
TITLE 13
TMP 11
TT-Option 16

Ueberlagerungs
-möglichkeiten 9
-segmente 10
-struktur 10, 22
Ueberlauf 42
UNDERFLOW 39
UNIT
- Nummer 27, 31
- Parameter 28, 31
UNIT005 28
UNIT006 28
Unterbrechungsanforderung 34

V-Option 15
VALUE-CONTROL 14
VC-Option 16
Verbinden von FORTRAN77-Programmen 23

Vorzeichenbit 33

Warnung 41
WPASS 9
W1PASS 9
W2PASS 9

X 27
X-Option 15
XR-Option 16
XREF 12

Y 27

Z 27
Zeichen "D" 12
Zeilenlänge 12
ZERODIVIDE 39
Zugriffsart 30